



---

Junos<sup>®</sup> OS

## Layer 3 VPNs Feature Guide for Routing Devices

Release

14.1



---

Published: 2014-12-01

Juniper Networks, Inc.  
1194 North Mathilda Avenue  
Sunnyvale, California 94089  
USA  
408-745-2000  
[www.juniper.net](http://www.juniper.net)

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

*Junos<sup>®</sup> OS Layer 3 VPNs Feature Guide for Routing Devices*

14.1

Copyright © 2014, Juniper Networks, Inc.  
All rights reserved.

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 . . . . .	xvii
	Documentation and Release Notes . . . . .	xvii
	Supported Platforms . . . . .	xvii
	Using the Examples in This Manual . . . . .	xvii
	Merging a Full Example . . . . .	xviii
	Merging a Snippet . . . . .	xviii
	Documentation Conventions . . . . .	xix
	Documentation Feedback . . . . .	xxi
	Requesting Technical Support . . . . .	xxi
	Self-Help Online Tools and Resources . . . . .	xxi
	Opening a Case with JTAC . . . . .	xxii
<b>Part 1</b>	<b>Overview</b>	
<b>Chapter 1</b>	<b>Introduction to Layer 3 VPNs . . . . .</b>	<b>3</b>
	Layer 3 VPN Introduction . . . . .	3
	Layer 3 VPN Platform Support . . . . .	4
	Layer 3 VPN Attributes . . . . .	4
	VPN-IPv4 Addresses and Route Distinguishers . . . . .	5
	IPv6 Layer 3 VPNs . . . . .	8
	Virtual Routing and Forwarding Tables . . . . .	8
	Route Distribution Within a Layer 3 VPN . . . . .	12
	Distribution of Routes from CE to PE Routers . . . . .	12
	Distribution of Routes Between PE Routers . . . . .	13
	Distribution of Routes from PE to CE Routers . . . . .	15
	Forwarding Across the Provider's Core Network . . . . .	16
	Routing Instances for VPNs . . . . .	17
	Multicast over Layer 3 VPNs . . . . .	17
	Multicast over Layer 3 VPNs Overview . . . . .	17
	Sending PIM Hello Messages to the PE Routers . . . . .	19
	Sending PIM Join Messages to the PE Routers . . . . .	20
	Receiving the Multicast Transmission . . . . .	20
	VPN Per-Packet Load Balancing . . . . .	21
<b>Chapter 2</b>	<b>Introduction to Configuring Layer 3 VPNs . . . . .</b>	<b>23</b>
	Configuring a VPN Tunnel for VRF Table Lookup . . . . .	23

## Part 2

### Chapter 3

## Configuration

<b>Configuring Layer 3 VPNs</b>	<b>27</b>
Introduction to Configuring Layer 3 VPNs	28
Configuring Routing Between PE and CE Routers in Layer 3 VPNs	31
Configuring BGP Between the PE and CE Routers	31
Configuring OSPF Between the PE and CE Routers	32
Configuring OSPF Version 2 Between the PE and CE Routers	32
Configuring OSPF Version 3 Between the PE and CE Routers	32
Configuring RIP Between the PE and CE Routers	33
Configuring Static Routes Between the PE and CE Routers	34
Example: Configuring OSPFv2 Sham Links	35
OSPFv2 Sham Links Overview	35
Example: Configuring OSPFv2 Sham Links	36
Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs	44
Configuring Layer 3 VPNs to Carry IPv6 Traffic	46
Configuring IPv6 on the PE Router	46
Configuring the Connection Between the PE and CE Routers	46
Configuring BGP on the PE Router to Handle IPv6 Routes	47
Configuring BGP on the PE Router for IPv4 and IPv6 Routes	47
Configuring OSPF Version 3 on the PE Router	48
Configuring Static Routes on the PE Router	48
Configuring IPv6 on the Interfaces	48
Configuring EBGP Multihop Sessions Between PE and CE Routers in Layer 3 VPNs	49
Configuring Layer 3 VPNs to Carry IBGP Traffic	49
Filtering Packets in Layer 3 VPNs Based on IP Headers	51
Egress Filtering Options	52
Support on Aggregated and VLAN Interfaces for IP-Based Filtering	53
Support on ATM and Frame Relay Interfaces for IP-Based Filtering	53
Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering	54
Support on SONET/SDH and DS3/E3 Channelized Enhanced Intelligent Queuing Interfaces for IP-Based Filtering	54
Support on Multilink PPP and Multilink Frame Relay Interfaces for IP-Based Filtering	56
Support for IP-Based Filtering of Packets with Null Top Labels	56
General Limitations on IP-Based Filtering	57
Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs	58
Load Balancing and IP Header Filtering for Layer 3 VPNs	59
Example: Configuring Provider Edge Link Protection in Layer 3 VPNs	59
Understanding Provider Edge Link Protection in Layer 3 VPNs	60
Example: Configuring Provider Edge Link Protection in Layer 3 VPNs	61
Understanding Provider Edge Link Protection for BGP Labeled Unicast Paths	76
Egress Protection for BGP Labeled Unicast	77
Configuring Egress Protection for BGP Labeled Unicast	79
Configuring a Label Allocation and Substitution Policy for VPNs	81

Configuring Logical Units on the Loopback Interface for Routing Instances in Layer 3 VPNs . . . . .	82
Configuring Multicast Layer 3 VPNs . . . . .	84
Configuring Packet Forwarding for Layer 3 VPNs . . . . .	85
Configuring GRE Tunnels for Layer 3 VPNs . . . . .	86
Configuring GRE Tunnels Manually Between PE and CE Routers . . . . .	87
Configuring the GRE Tunnel Interface on the PE Router . . . . .	87
Configuring the GRE Tunnel Interface on the CE Router . . . . .	88
Configuring GRE Tunnels Dynamically . . . . .	89
Configuring an ES Tunnel Interface for Layer 3 VPNs . . . . .	90
Configuring the ES Tunnel Interface on the PE Router . . . . .	90
Configuring the ES Tunnel Interface on the CE Router . . . . .	91
Configuring IPsec Tunnels Instead of MPLS LSPs Between PE Routers in Layer 3 VPNs . . . . .	92
Configuring Protocol-Independent Load Balancing in Layer 3 VPNs . . . . .	95
Configuring Load Balancing for Layer 3 VPNs . . . . .	95
Configuring Load Balancing and Routing Policies . . . . .	96
Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes . . . . .	97
Configuring Traffic Policing in Layer 3 VPNs . . . . .	98
Configuring BGP PIC Edge for MPLS Layer 3 VPNs . . . . .	99
Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs . . . . .	101
Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs . . . . .	112
Accepting Up to One Million Layer 3 VPN Route Updates . . . . .	112
Accepting More Than One Million Layer 3 VPN Route Updates . . . . .	113
Enabling Chained Composite Next Hops for IPv6 Labeled Unicast Routes . . . . .	115
<b>Chapter 4 Layer 3 VPN Configuration Examples . . . . .</b>	<b>117</b>
Configuring a Simple Full-Mesh VPN Topology . . . . .	118
Enabling an IGP on the PE and P Routers . . . . .	119
Enabling RSVP and MPLS on the P Router . . . . .	119
Configuring the MPLS LSP Tunnel Between the PE Routers . . . . .	120
Configuring IBGP on the PE Routers . . . . .	121
Configuring Routing Instances for VPNs on the PE Routers . . . . .	122
Configuring VPN Policy on the PE Routers . . . . .	124
Simple VPN Configuration Summarized by Router . . . . .	127
Router A (PE Router) . . . . .	127
Router B (P Router) . . . . .	129
Router C (PE Router) . . . . .	130
Configuring a Full-Mesh VPN Topology with Route Reflectors . . . . .	132
Configuring Hub-and-Spoke VPN Topologies: One Interface . . . . .	132
Configuring Hub CE1 . . . . .	134
Configuring Hub PE1 . . . . .	135
Configuring the P Router . . . . .	135
Configuring Spoke PE2 . . . . .	136
Configuring Spoke PE3 . . . . .	137
Configuring Spoke CE2 . . . . .	138
Configuring Spoke CE3 . . . . .	139

Enabling Egress Features on the Hub PE Router . . . . .	141
Configuring Hub PE1 . . . . .	141
Configuring Hub-and-Spoke VPN Topologies: Two Interfaces . . . . .	144
Enabling an IGP on the Hub-and-Spoke PE Routers . . . . .	147
Configuring LDP on the Hub-and-Spoke PE Routers . . . . .	147
Configuring IBGP on the PE Routers . . . . .	148
Configuring VPN Routing Instances on the Hub-and-Spoke PE Routers . . . . .	149
Configuring VPN Policy on the PE Routers . . . . .	151
Hub-and-Spoke VPN Configuration Summarized by Router . . . . .	154
Router D (Hub PE Router) . . . . .	154
Router E (Spoke PE Router) . . . . .	156
Router F (Spoke PE Router) . . . . .	157
Configuring an LDP-over-RSVP VPN Topology . . . . .	159
Enabling an IGP on the PE and P Routers . . . . .	162
Enabling LDP on the PE and P Routers . . . . .	162
Enabling RSVP and MPLS on the P Router . . . . .	164
Configuring the MPLS LSP Tunnel Between the P Routers . . . . .	164
Configuring IBGP on the PE Routers . . . . .	165
Configuring Routing Instances for VPNs on the PE Routers . . . . .	166
Configuring VPN Policy on the PE Routers . . . . .	167
LDP-over-RSVP VPN Configuration Summarized by Router . . . . .	169
Router PE1 . . . . .	169
Router P1 . . . . .	170
Router P2 . . . . .	171
Router P3 . . . . .	171
Router PE2 . . . . .	172
Configuring an Application-Based Layer 3 VPN Topology . . . . .	173
Configuration on Router A . . . . .	175
Configuration on Router E . . . . .	176
Configuration on Router F . . . . .	177
Configuring an OSPF Domain ID for a Layer 3 VPN . . . . .	178
Configuring Interfaces on Router PE1 . . . . .	178
Configuring Routing Options on Router PE1 . . . . .	179
Configuring Protocols on Router PE1 . . . . .	179
Configuring Policy Options on Router PE1 . . . . .	180
Configuring the Routing Instance on Router PE1 . . . . .	180
Configuration Summary for Router PE1 . . . . .	181
Configuring Overlapping VPNs Using Routing Table Groups . . . . .	183
Configuring Routing Table Groups . . . . .	184
Configuring Static Routes Between the PE and CE Routers . . . . .	185
Configuring the Routing Instance for VPN A . . . . .	185
Configuring the Routing Instance for VPN AB . . . . .	186
Configuring the Routing Instance for VPN B . . . . .	186
Configuring VPN Policy . . . . .	187
Configuring BGP Between the PE and CE Routers . . . . .	190
Configuring OSPF Between the PE and CE Routers . . . . .	191
Configuring Static, BGP, and OSPF Routes Between PE and CE Routers . . . . .	193

Configuring Overlapping VPNs Using Automatic Route Export . . . . .	194
Configuring Overlapping VPNs with BGP and Automatic Route Export . . . .	195
Configuring Overlapping VPNs and Additional Tables . . . . .	196
Configuring Automatic Route Export for All VRF Instances . . . . .	197
Configuring a GRE Tunnel Interface Between PE Routers . . . . .	198
Configuring the Routing Instance on Router A . . . . .	198
Configuring the Routing Instance on Router D . . . . .	199
Configuring MPLS, BGP, and OSPF on Router A . . . . .	199
Configuring MPLS, BGP, and OSPF on Router D . . . . .	200
Configuring the Tunnel Interface on Router A . . . . .	200
Configuring the Tunnel Interface on Router D . . . . .	201
Configuring the Routing Options on Router A . . . . .	201
Configuring the Routing Options on Router D . . . . .	201
Configuration Summary for Router A . . . . .	202
Configuration Summary for Router D . . . . .	203
Configuring a GRE Tunnel Interface Between a PE and CE Router . . . . .	204
Configuring the Routing Instance Without the Encapsulating Interface . . .	205
Configuring the Routing Instance on Router PE1 . . . . .	205
Configuring the GRE Tunnel Interface on Router PE1 . . . . .	205
Configuring the Encapsulation Interface on Router PE1 . . . . .	206
Configuring the Routing Instance with the Encapsulating Interface . . . . .	206
Configuring the Routing Instance on Router PE1 . . . . .	206
Configuring the GRE Tunnel Interface on Router PE1 . . . . .	207
Configuring the Encapsulation Interface on Router PE1 . . . . .	208
Configuring the GRE Tunnel Interface on Router CE1 . . . . .	208
Configuring an ES Tunnel Interface Between a PE and CE Router . . . . .	208
Configuring IPsec on Router PE1 . . . . .	209
Configuring the Routing Instance Without the Encapsulating Interface . . .	209
Configuring the Routing Instance on Router PE1 . . . . .	209
Configuring the ES Tunnel Interface on Router PE1 . . . . .	210
Configuring the Encapsulating Interface for the ES Tunnel . . . . .	210
Configuring the Routing Instance with the Encapsulating Interface . . . . .	210
Configuring the Routing Instance on Router PE1 . . . . .	210
Configuring the ES Tunnel Interface on Router PE1 . . . . .	211
Configuring the Encapsulating Interface on Router PE1 . . . . .	211
Configuring the ES Tunnel Interface on Router CE1 . . . . .	211
Configuring IPsec on Router CE1 . . . . .	212
Layer 3 VPN Load Balancing Using IP Header Filtering . . . . .	212
Layer 3 VPN Load Balancing Overview . . . . .	212
Example: Load Balancing Layer 3 VPN Traffic While Simultaneously Using IP Header Filtering . . . . .	213
Example: Disabling Normal TTL Decrementing in a VRF Routing Instance . . . .	227
Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters . . . . .	234
Example: Configuring Route Resolution on PE Routers . . . . .	238

Example: Configuring Route Resolution on Route Reflectors . . . . .	239
Example: Configuring Link Protection with Host Fast Reroute . . . . .	242
Understanding Host Fast Reroute . . . . .	242
ARP Prefix Limit and Blackout Supplementary Timeout . . . . .	244
Primary Route and Backup Route Candidates . . . . .	245
Backup Path Selection Policy . . . . .	245
Characteristics of HFRR Routes . . . . .	246
Removal of HFRR Routes . . . . .	246
Interfaces That Support HFRR . . . . .	246
Example: Configuring Link Protection with Host Fast Reroute . . . . .	247
Example: Configuring a Layer 3 VPN with Route Reflection and AS Override . . . . .	258
Example: Configuring MPLS Egress Protection for Layer 3 VPN Services . . . . .	268
Egress Protection for Layer 3 VPN Edge Protection Overview . . . . .	269
Router Functions . . . . .	270
Protector and Protection Models . . . . .	271
IGP Advertisement Model . . . . .	271
Example: Configuring Egress Protection for Layer 3 VPN Services . . . . .	275
Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP . . . . .	283
Example: Configuring Provider Edge Link Protection in Layer 3 VPNs . . . . .	316
Understanding Provider Edge Link Protection in Layer 3 VPNs . . . . .	316
Example: Configuring Provider Edge Link Protection in Layer 3 VPNs . . . . .	317
Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths . . . . .	332
Example: Configuring Egress Protection for BGP Labeled Unicast . . . . .	345
Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains . . . . .	358
<b>Chapter 5 Layer 3 VPN Internet Access Examples . . . . .</b>	<b>369</b>
Non-VRF Internet Access . . . . .	369
CE Router Accesses Internet Independently of the PE Router . . . . .	370
PE Router Provides Layer 2 Internet Service . . . . .	370
Distributed Internet Access . . . . .	370
Routing VPN and Internet Traffic Through Different Interfaces . . . . .	371
Configuring Interfaces on Router PE1 . . . . .	372
Configuring Routing Options on Router PE1 . . . . .	372
Configuring BGP, IS-IS, and LDP Protocols on Router PE1 . . . . .	372
Configuring a Routing Instance on Router PE1 . . . . .	373
Configuring Policy Options on Router PE1 . . . . .	374
Traffic Routed by Different Interfaces: Configuration Summarized by Router . . . . .	375
Router PE1 . . . . .	375
Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface . . . . .	377
Configuration for Router PE1 . . . . .	377
Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses) . . . . .	378
Configuring Routing Options on Router PE1 . . . . .	379
Configuring Routing Protocols on Router PE1 . . . . .	379
Configuring the Routing Instance on Router PE1 . . . . .	380

	Traffic Routed Through the Same Interface Bidirectionally: Configuration Summarized by Router . . . . .	381
	Router PE1 . . . . .	381
	Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Private Addresses) . . . . .	382
	Configuring Routing Options for Router PE1 . . . . .	383
	Configuring a Routing Instance for Router PE1 . . . . .	383
	Configuring Policy Options for Router PE1 . . . . .	384
	Traffic Routed by the Same Interface Bidirectionally (VPN Has Private Addresses): Configuration Summarized by Router . . . . .	384
	Router PE1 . . . . .	384
	Routing Internet Traffic Through a Separate NAT Device . . . . .	386
	Centralized Internet Access . . . . .	394
	Routing Internet Traffic Through a Hub CE Router . . . . .	394
	Configuring a Routing Instance on Router PE1 . . . . .	395
	Configuring Policy Options on Router PE1 . . . . .	396
	Internet Traffic Routed by a Hub CE Router: Configuration Summarized by Router . . . . .	397
	Routing Internet Traffic Through Multiple CE Routers . . . . .	398
	Configuring a Routing Instance on Router PE1 . . . . .	399
	Configuring Policy Options on Router PE1 . . . . .	399
	Configuring a Routing Instance on Router PE3 . . . . .	400
	Configuring Policy Options on Router PE3 . . . . .	401
	Routing Internet Traffic Through Multiple CE Routers: Configuration Summarized by Router . . . . .	402
<b>Chapter 6</b>	<b>Layer 3 VPN Additional Examples . . . . .</b>	<b>405</b>
	Example: Configuring Interprovider Layer 3 VPN Option A . . . . .	405
	Example: Configuring Interprovider Layer 3 VPN Option B . . . . .	425
	Example: Configuring Interprovider Layer 3 VPN Option C . . . . .	444
	Layer 3 VPN Overview . . . . .	466
	Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN . . . . .	468
	Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview . . . . .	468
	Interconnecting Layer 2 VPNs with Layer 3 VPNs Applications . . . . .	469
	Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN . . . . .	470
	Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN . . . . .	493
<b>Chapter 7</b>	<b>Summary of Layer 3 VPN Configuration Statements . . . . .</b>	<b>513</b>
	advertise-mode (MPLS) . . . . .	515
	arp-prefix-limit (Host Fast Reroute) . . . . .	516
	as-path-compare . . . . .	517
	chained-composite-next-hop . . . . .	518
	classifiers . . . . .	519
	description (Routing Instances) . . . . .	519
	domain-id . . . . .	520
	domain-vpn-tag . . . . .	520
	dynamic-tunnels . . . . .	521
	egress-protection (MPLS) . . . . .	522
	egress-protection (Routing Instances) . . . . .	523
	egress-protection (BGP) . . . . .	524

	extended-space . . . . .	525
	forwarding-table . . . . .	526
	global-arp-prefix-limit (Host Fast Reroute) . . . . .	527
	global-supplementary-blackout-timer (Host Fast Reroute) . . . . .	529
	group-address (Routing Instances VPN) . . . . .	531
	host-fast-reroute . . . . .	532
	independent-domain . . . . .	533
	ingress (Chained Composite Next Hop) . . . . .	534
	inet6 . . . . .	535
	inet6-vpn . . . . .	536
	interface (Host Fast Reroute) . . . . .	537
	l3vpn . . . . .	538
	l3vpn (transit) . . . . .	539
	label . . . . .	539
	labeled-bgp . . . . .	540
	link-protection (Host Fast Reroute) . . . . .	540
	maximum-paths . . . . .	541
	maximum-prefixes . . . . .	543
	metric (Protocols OSPF Sham Link) . . . . .	544
	multihop . . . . .	545
	multipath (Routing Options) . . . . .	546
	no-vrf-advertise . . . . .	547
	no-vrf-propagate-ttl . . . . .	548
	protect core . . . . .	549
	protection (Protocols BGP) . . . . .	549
	routing-instances (Class of Service) . . . . .	550
	sham-link . . . . .	551
	sham-link-remote . . . . .	552
	source-class-usage . . . . .	553
	supplementary-blackout-timer (Host Fast Reroute) . . . . .	554
	vpn-unequal-cost . . . . .	555
	vrf-propagate-ttl . . . . .	556
	vrf-table-label . . . . .	557
<b>Part 3</b>	<b>Administration</b>	
<b>Chapter 8</b>	<b>Layer 3 VPNs Reference . . . . .</b>	<b>561</b>
	Supported Layer 3 VPN Standards . . . . .	561
<b>Chapter 9</b>	<b>Operational Commands . . . . .</b>	<b>563</b>
	ping mpls l3vpn . . . . .	564
	show hfr profiles . . . . .	567
<b>Part 4</b>	<b>Troubleshooting</b>	
<b>Chapter 10</b>	<b>Troubleshooting Layer 3 VPNs . . . . .</b>	<b>571</b>
	Diagnosing Common Problems . . . . .	571
	Example: Troubleshooting Layer 3 VPNs . . . . .	574

## Part 5

## Index

Index .....	587
-------------	-----



# List of Figures

<b>Part 1</b>	<b>Overview</b>	
<b>Chapter 1</b>	<b>Introduction to Layer 3 VPNs</b>	<b>3</b>
	Figure 1: VPN Attributes and Route Distribution	5
	Figure 2: Overlapping Addresses Among Different VPNs	6
	Figure 3: Route Distinguishers	8
	Figure 4: VRF Tables	9
	Figure 5: Route Distribution Within a VPN	12
	Figure 6: Distribution of Routes from CE Routers to PE Routers	13
	Figure 7: Distribution of Routes Between PE Routers	14
	Figure 8: Distribution of Routes from PE Routers to CE Routers	15
	Figure 9: Using MPLS LSPs to Tunnel Between PE Routers	16
	Figure 10: Label Stack	17
	Figure 11: Multicast Topology Overview	19
<b>Part 2</b>	<b>Configuration</b>	
<b>Chapter 3</b>	<b>Configuring Layer 3 VPNs</b>	<b>27</b>
	Figure 12: OSPFv2 Sham Link	35
	Figure 13: OSPFv2 Sham Link Example	37
	Figure 14: Provider Edge Link Protection in a Layer 3 VPN	62
	Figure 15: Inter-AS Option C	78
	Figure 16: Seamless MPLS	78
	Figure 17: GRE Tunnel Configured Between the Local CE Router and the PE Router	87
	Figure 18: GRE Tunnel Configured Between the Remote CE Router and the PE Router	87
	Figure 19: BGP PIC Edge Scenario	102
<b>Chapter 4</b>	<b>Layer 3 VPN Configuration Examples</b>	<b>117</b>
	Figure 20: Example of a Simple VPN Topology	118
	Figure 21: Example of a Hub-and-Spoke VPN Topology with One Interface	133
	Figure 22: Example of a Hub-and-Spoke VPN Topology with Two Interfaces	145
	Figure 23: Route Distribution Between Two Spoke Routers	146
	Figure 24: Example of an LDP-over-RSVP VPN Topology	159
	Figure 25: Label Pushing and Popping	161
	Figure 26: Application-Based Layer 3 VPN Example Configuration	174
	Figure 27: Example of a Configuration Using an OSPF Domain ID	178
	Figure 28: Example of an Overlapping VPN Topology	184
	Figure 29: PE Routers A and D Connected by a GRE Tunnel Interface	198
	Figure 30: GRE Tunnel Between the CE Router and the PE Router	204

	Figure 31: ES Tunnel Interface (IPsec Tunnel) . . . . .	208
	Figure 32: Layer 3 VPN Load Balancing Using IP Header Filtering . . . . .	214
	Figure 33: Disabling TTL Propagation for a Single VPN . . . . .	229
	Figure 34: Host Fast Reroute . . . . .	243
	Figure 35: Host Fast Reroute Topology . . . . .	249
	Figure 36: AS Override Topology . . . . .	259
	Figure 37: Sample Topology for Egress Protection . . . . .	269
	Figure 38: Layer 3 VPN Egress Protection with RSVP and LDP . . . . .	285
	Figure 39: Provider Edge Link Protection in a Layer 3 VPN . . . . .	318
	Figure 40: Labeled Unicast Link Protection in a Layer 3 VPN . . . . .	333
	Figure 41: Egress Protection in a Layer 3 VPN . . . . .	347
	Figure 42: Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains . . . . .	360
<b>Chapter 5</b>	<b>Layer 3 VPN Internet Access Examples . . . . .</b>	<b>369</b>
	Figure 43: PE Router Does Not Provide Internet Access . . . . .	370
	Figure 44: PE Router Connects to a Router Connected to the Internet . . . . .	370
	Figure 45: Routing VPN and Internet Traffic Through Different Interfaces . . . . .	371
	Figure 46: Example of Internet Traffic Routed Through Separate Interfaces . . . . .	371
	Figure 47: VPN and Outgoing Internet Traffic Routed Through the Same Interface and Return Internet Traffic Routed Through a Different Interface . . . . .	377
	Figure 48: Interface Configured to Carry Both Internet and VPN Traffic . . . . .	378
	Figure 49: VPN and Internet Traffic Routed Through the Same Interface . . . . .	382
	Figure 50: Internet Traffic Routed Through a Separate NAT Device . . . . .	386
	Figure 51: Internet Traffic Routed Through a NAT Example Topology . . . . .	387
	Figure 52: Internet Access Through a Hub CE Router Performing NAT . . . . .	394
	Figure 53: Internet Access Provided Through a Hub CE Router . . . . .	395
	Figure 54: Two Hub CE Routers Handling Internet Traffic and NAT . . . . .	398
<b>Chapter 6</b>	<b>Layer 3 VPN Additional Examples . . . . .</b>	<b>405</b>
	Figure 55: Physical Topology of Interprovider Layer 3 VPN Option A . . . . .	407
	Figure 56: Physical Topology of Interprovider Layer 3 VPN Option B . . . . .	427
	Figure 57: Physical Topology of Interprovider Layer 3 VPN Option C . . . . .	447
	Figure 58: Physical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN . . . . .	472
	Figure 59: Logical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN . . . . .	472
	Figure 60: Physical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection . . . . .	494
	Figure 61: Logical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection . . . . .	494
<b>Part 4</b>	<b>Troubleshooting</b>	
<b>Chapter 10</b>	<b>Troubleshooting Layer 3 VPNs . . . . .</b>	<b>571</b>
	Figure 62: Layer 3 VPN Topology for ping and traceroute Examples . . . . .	576

# List of Tables

	<b>About the Documentation</b> . . . . .	<b>xvii</b>
	Table 1: Notice Icons . . . . .	xix
	Table 2: Text and Syntax Conventions . . . . .	xx
<b>Part 1</b>	<b>Overview</b>	
<b>Chapter 1</b>	<b>Introduction to Layer 3 VPNs</b> . . . . .	<b>3</b>
	Table 3: Ingress Router Hashing . . . . .	21
	Table 4: Transit and Egress Router Hashing . . . . .	22
<b>Part 2</b>	<b>Configuration</b>	
<b>Chapter 3</b>	<b>Configuring Layer 3 VPNs</b> . . . . .	<b>27</b>
	Table 5: Support for Aggregated and VLAN Interfaces . . . . .	53
	Table 6: Support for ATM and Frame Relay Interfaces . . . . .	53
	Table 7: Support for Ethernet, SONET/SDH, and T1/T3/E3 Interfaces . . . . .	54
	Table 8: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs . . . . .	54
	Table 9: Support for Multilink PPP and Multilink Frame Relay Interfaces . . . . .	56
<b>Chapter 4</b>	<b>Layer 3 VPN Configuration Examples</b> . . . . .	<b>117</b>
	Table 10: Device IP Address Quick Reference . . . . .	215
<b>Part 3</b>	<b>Administration</b>	
<b>Chapter 9</b>	<b>Operational Commands</b> . . . . .	<b>563</b>
	Table 11: show hfr profiles Output Fields . . . . .	567



# About the Documentation

- [Documentation and Release Notes on page xvii](#)
- [Supported Platforms on page xvii](#)
- [Using the Examples in This Manual on page xvii](#)
- [Documentation Conventions on page xix](#)
- [Documentation Feedback on page xxi](#)
- [Requesting Technical Support on page xxi](#)

## 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>.

## Supported Platforms

---

For the features described in this document, the following platforms are supported:

- [MX Series](#)
- [T Series](#)
- [M Series](#)
- [PTX Series](#)

## 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.xml;
    }
  }
}
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.xml; }
```

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 *CLI User Guide*.

## Documentation Conventions

Table 1 on page xix 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.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

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

Table 2: Text and Syntax Conventions

Convention	Description	Examples
<b>Bold text like this</b>	Represents text that you type.	To enter configuration mode, type the <b>configure</b> command:  user@host> <b>configure</b>
Fixed-width text like this	Represents output that appears on the terminal screen.	user@host> <b>show chassis alarms</b>  No alarms currently active
<i>Italic text like this</i>	<ul style="list-style-type: none"> <li>Introduces or emphasizes important new terms.</li> <li>Identifies guide names.</li> <li>Identifies RFC and Internet draft titles.</li> </ul>	<ul style="list-style-type: none"> <li>A policy <i>term</i> is a named structure that defines match conditions and actions.</li> <li><i>Junos OS CLI User Guide</i></li> <li>RFC 1997, <i>BGP Communities Attribute</i></li> </ul>
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name:  [edit] root@# <b>set system domain-name</b> <i>domain-name</i>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none"> <li>To configure a stub area, include the <b>stub</b> statement at the [edit protocols <b>ospf area area-id</b>] hierarchy level.</li> <li>The console port is labeled <b>CONSOLE</b>.</li> </ul>
< > (angle brackets)	Encloses optional keywords or variables.	<b>stub &lt;default-metric metric&gt;;</b>
(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.	<b>broadcast   multicast</b>  <b>(string1   string2   string3)</b>
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	<b>rsvp { # Required for dynamic MPLS only</b>
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	<b>community name members [ community-ids ]</b>
Indentation and braces ( { } )	Identifies a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	}

---

#### GUI Conventions

---

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

Convention	Description	Examples
<b>Bold text like this</b>	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none"> <li>In the Logical Interfaces box, select <b>All Interfaces</b>.</li> <li>To cancel the configuration, click <b>Cancel</b>.</li> </ul>
<b>&gt;</b> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select <b>Protocols&gt;Ospf</b> .

## Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page at the Juniper Networks Technical Documentation site at <http://www.juniper.net/techpubs/index.html>, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at <https://www.juniper.net/cgi-bin/docbugreport/>.
- E-mail—Send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net). Include the document or topic name, URL or page number, and software 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: <http://kb.juniper.net/InfoCenter/>
- 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 Layer 3 VPNs on page 3](#)
- [Introduction to Configuring Layer 3 VPNs on page 23](#)



## CHAPTER 1

# Introduction to Layer 3 VPNs

- [Layer 3 VPN Introduction on page 3](#)
- [Layer 3 VPN Platform Support on page 4](#)
- [Layer 3 VPN Attributes on page 4](#)
- [VPN-IPv4 Addresses and Route Distinguishers on page 5](#)
- [IPv6 Layer 3 VPNs on page 8](#)
- [Virtual Routing and Forwarding Tables on page 8](#)
- [Route Distribution Within a Layer 3 VPN on page 12](#)
- [Forwarding Across the Provider's Core Network on page 16](#)
- [Routing Instances for VPNs on page 17](#)
- [Multicast over Layer 3 VPNs on page 17](#)
- [VPN Per-Packet Load Balancing on page 21](#)

## Layer 3 VPN Introduction

---

In Junos OS, Layer 3 VPNs are based on RFC 4364. RFC 4364 defines a mechanism by which service providers can use their IP backbones to provide VPN services to their customers. A Layer 3 VPN is a set of sites that share common routing information and whose connectivity is controlled by a collection of policies. The sites that make up a Layer 3 VPN are connected over a provider's existing public Internet backbone.

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

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

## Layer 3 VPN Platform Support

---

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

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

## Layer 3 VPN Attributes

---

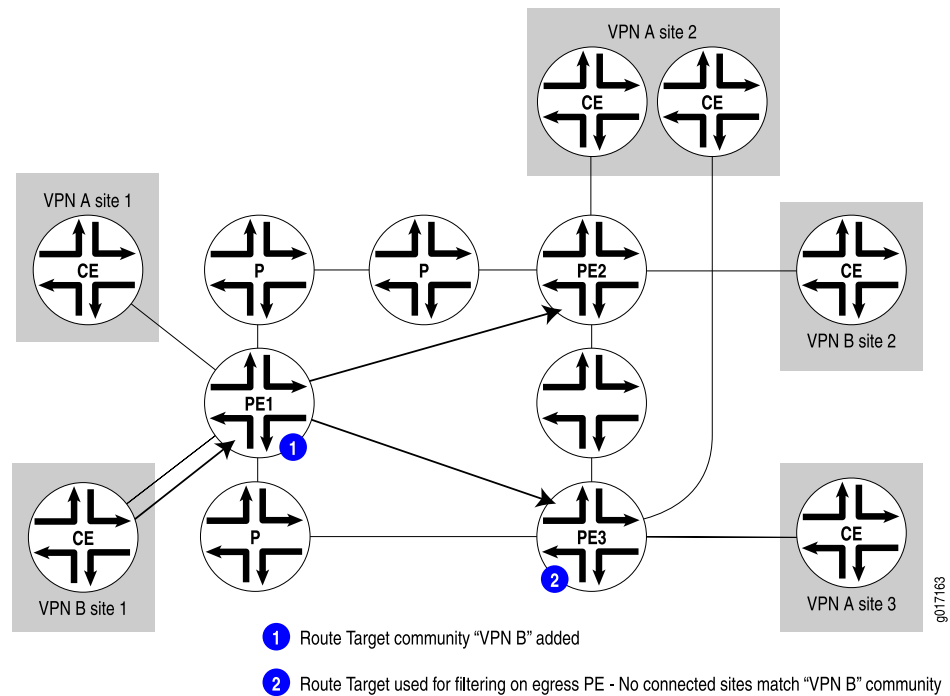
Route distribution within a VPN is controlled through BGP extended community attributes. RFC 4364 defines the following three attributes used by VPNs:

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

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

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

Figure 1: VPN Attributes and Route Distribution

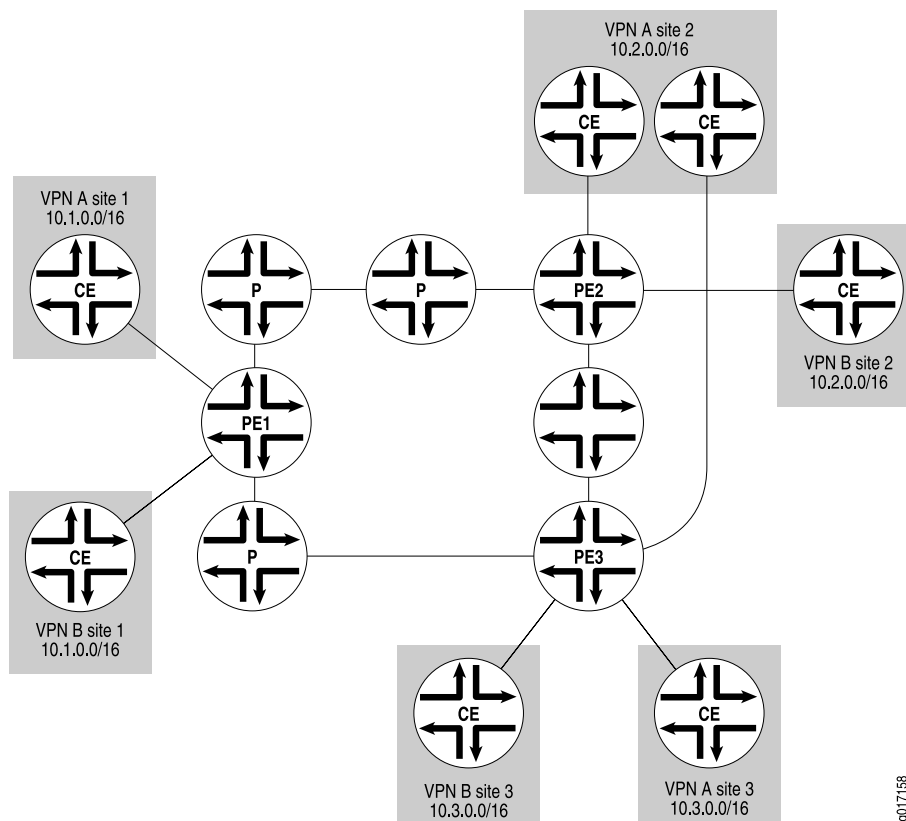


## VPN-IPv4 Addresses and Route Distinguishers

Because Layer 3 VPNs connect private networks—which can use either public addresses or private addresses, as defined in RFC 1918 (*Address Allocation for Private Internets*)—over the public Internet infrastructure, when the private networks use private addresses, the addresses might overlap with the addresses of another private network.

Figure 2 on page 6 illustrates how private addresses of different private networks can overlap. Here, sites within VPN A and VPN B use the address spaces 10.1.0.0/16, 10.2.0.0/16, and 10.3.0.0/16 for their private networks.

Figure 2: Overlapping Addresses Among Different VPNs



g017158

To avoid overlapping private addresses, you can configure the network devices to use public addresses instead of private addresses. However, this is a large and complex undertaking. The solution provided in RFC 4364 uses the existing private network numbers to create a new address that is unambiguous. The new address is part of the VPN-IPv4 address family, which is a BGP address family added as an extension to the BGP protocol. In VPN-IPv4 addresses, a value that identifies the VPN, called a route distinguisher, is prefixed to the private IPv4 address, providing an address that uniquely identifies a private IPv4 address.

Only the PE routers need to support the VPN-IPv4 address extension to BGP. When an ingress PE router receives an IPv4 route from a device within a VPN, it converts it into a VPN-IPv4 route by adding the route distinguisher prefix to the route. The VPN-IPv4 addresses are used only for routes exchanged between PE routers. When an egress PE router receives a VPN-IPv4 route, it converts the VPN-IPv4 route back to an IPv4 route by removing the route distinguisher before announcing the route to its connected CE routers.

VPN-IPv4 addresses have the following format:

- Route distinguisher is a 6-byte value that you can specify in one of the following formats:
  - **as-number:number**, where **as-number** is an AS number (a 2-byte value) and **number** is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned,

nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.

- ***ip-address:number***, where ***ip-address*** is an IP address (a 4-byte value) and ***number*** is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the **router-id** statement, which is a nonprivate address in your assigned prefix range.
- IPv4 address—4-byte address of a device within the VPN.

Figure 2 on page 6 illustrates how the AS number can be used in the route distinguisher. Suppose that VPN A is in AS 65535 and that VPN B is in AS 666 (both these AS numbers belong to the ISP), and suppose that the route distinguisher for Site 2 in VPN A is 65535:02 and that the route distinguisher for Site 2 in VPN B is 666:02. When Router PE2 receives a route from the CE router in VPN A, it converts it from its IP address of 10.2.0.0 to a VPN-IPv4 address of 65535:02:10.2.0.0. When the PE router receives a route from VPN B, which uses the same address space as VPN A, it converts it to a VPN-IPv4 address of 666:02:10.2.0.0.

If the IP address is used in the route distinguisher, suppose Router PE2's IP address is 172.168.0.1. When the PE router receives a route from VPN A, it converts it to a VPN-IPv4 address of 172.168.0.1:0:10.2.0.0/16, and it converts a route from VPN B to 172.168.0.0:1:10.2.0.0/16.

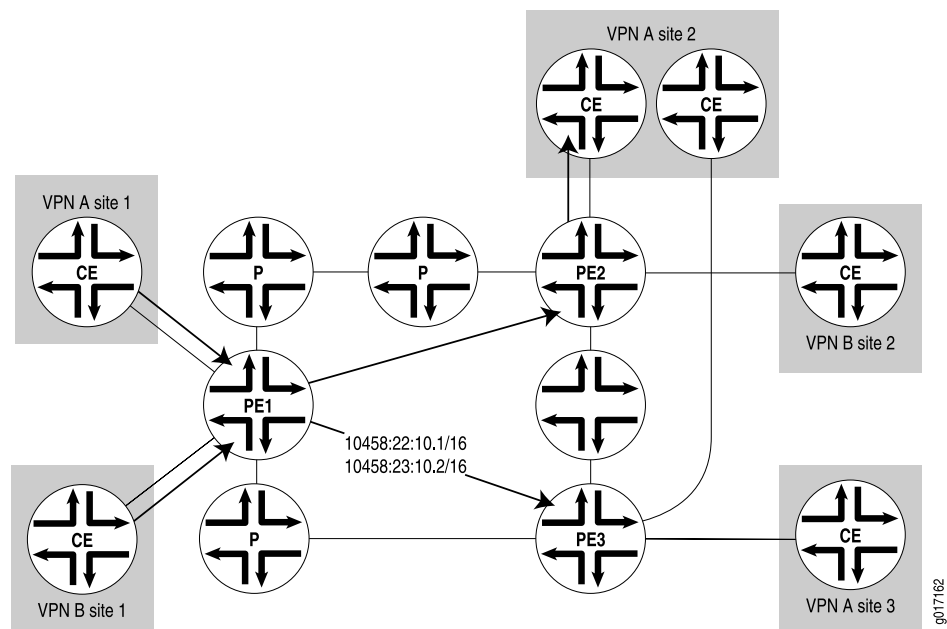
Route distinguishers are used only among PE routers to IPv4 addresses from different VPNs. The ingress PE router creates a route distinguisher and converts IPv4 routes received from CE routers into VPN-IPv4 addresses. The egress PE routers convert VPN-IPv4 routes into IPv4 routes before announcing them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

Because VPN-IPv4 addresses are a type of BGP address, you must configure IBGP sessions between pairs of PE routers so that the PE routers can distribute VPN-IPv4 routes within the provider's core network. (All PE routers are assumed to be within the same AS.)

You define BGP communities to constrain the distribution of routes among the PE routers. Defining BGP communities does not, by itself, distinguish IPv4 addresses.

Figure 3 on page 8 illustrates how Router PE1 adds the route distinguisher 10458:22:10.1/16 to routes received from the CE router at Site 1 in VPN A and forwards these routes to the other two PE routers. Similarly, Router PE1 adds the route distinguisher 10458:23:10.2/16 to routes received by the CE router at Site 1 in VPN B and forwards these routes to the other PE routers.

Figure 3: Route Distinguishers



## IPv6 Layer 3 VPNs

The interfaces between the PE and CE routers of a Layer 3 VPN can be configured to carry IP version 6 (IPv6) traffic. IP allows numerous nodes on different networks to interoperate seamlessly. IPv4 is currently used in intranets and private networks, as well as the Internet. IPv6 is the successor to IPv4, and is based for the most part on IPv4.

In the Juniper Networks implementation of IPv6, the service provider implements an MPLS-enabled IPv4 backbone to provide VPN service for IPv6 customers. The PE routers have both IPv4 and IPv6 capabilities. They maintain IPv6 VPN routing and forwarding (VRF) tables for their IPv6 sites and encapsulate IPv6 traffic in MPLS frames that are then sent into the MPLS core network.

IPv6 for Layer 3 VPNs is supported for BGP and for static routes.

IPv6 over Layer 3 VPNs is described in RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*.

For more information about IPv6, see the *Junos OS Routing Protocols Library for Routing Devices*.

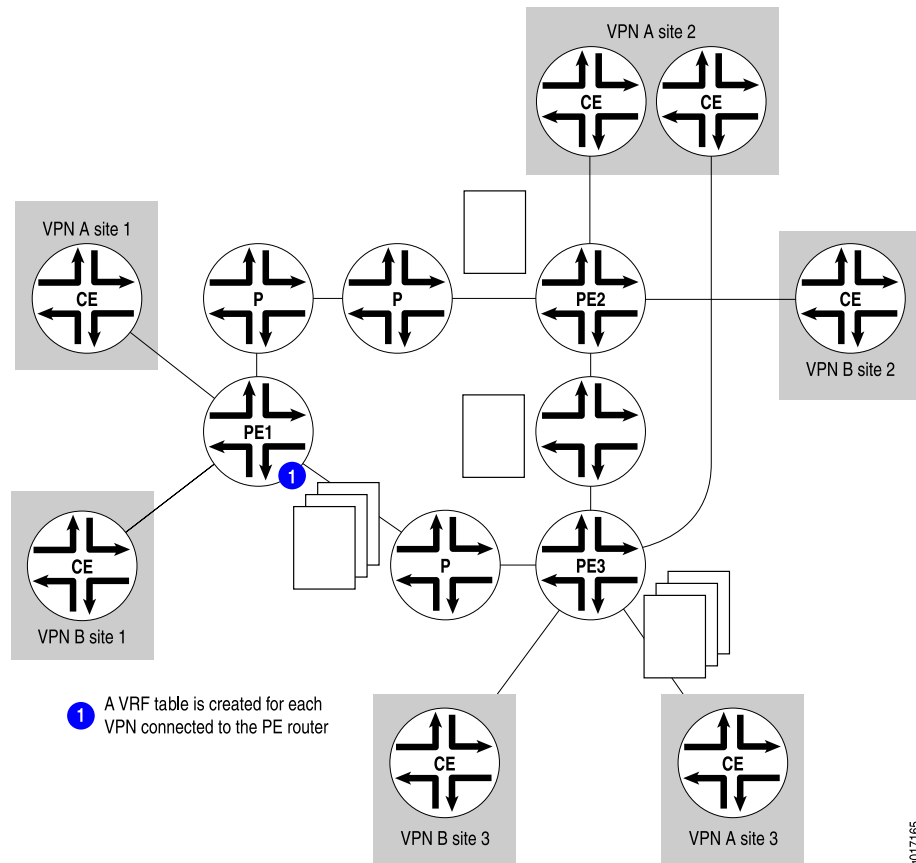
## Virtual Routing and Forwarding Tables

To separate a VPN's routes from routes in the public Internet or those in other VPNs, the PE router creates a separate routing table for each VPN, called a virtual routing and forwarding (VRF) table. The PE router creates one VRF table for each VPN that has a

connection to a CE router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN.

Figure 4 on page 9 illustrates the VRF tables that are created on the PE routers. The three PE routers have connections to CE routers that are in two different VPNs, so each PE router creates two VRF tables, one for each VPN.

Figure 4: VRF Tables



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

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

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

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

The Junos OS uses the following routing tables for VPNs:

- **bgp.l3vpn.0**—Stores all VPN-IPv4 unicast routes received from other PE routers. (This table does not store routes received from directly connected CE routers.) This table is present only on PE routers.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

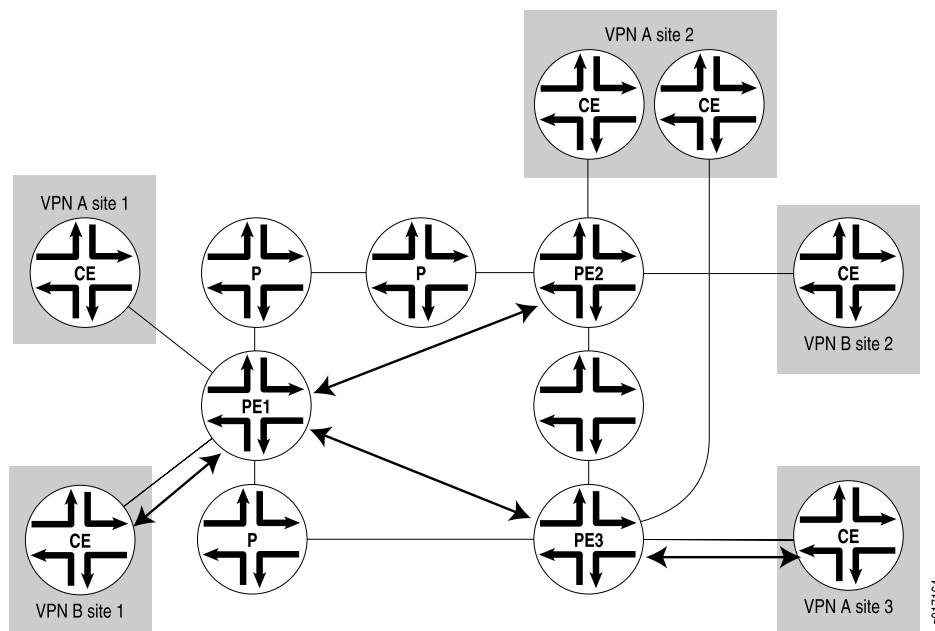
**Related Documentation**

- *IGP Shortcuts and VPNs*

## Route Distribution Within a Layer 3 VPN

Within a VPN, the distribution of VPN-IPv4 routes occurs between the PE and CE routers and between the PE routers (see [Figure 5 on page 12](#)).

Figure 5: Route Distribution Within a VPN



This section discusses the following topics:

- [Distribution of Routes from CE to PE Routers on page 12](#)
- [Distribution of Routes Between PE Routers on page 13](#)
- [Distribution of Routes from PE to CE Routers on page 15](#)

### Distribution of Routes from CE to PE Routers

A CE router announces its routes to the directly connected PE router. The announced routes are in IPv4 format. The PE router places the routes into the VRF table for the VPN. In the Junos OS, this is the *routing-instance-name.inet.0* routing table, where *routing-instance-name* is the configured name of the VPN.

The connection between the CE and PE routers can be a remote connection (a WAN connection) or a direct connection (such as a Frame Relay or Ethernet connection).

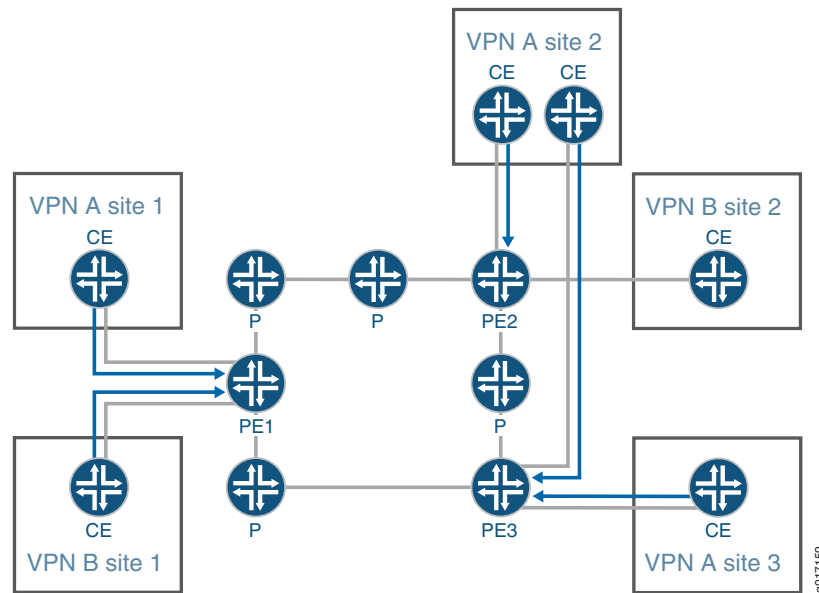
CE routers can communicate with PE routers using one of the following:

- OSPF
- RIP

- BGP
- Static route

Figure 6 on page 13 illustrates how routes are distributed from CE routers to PE routers. Router PE1 is connected to two CE routers that are in different VPNs. Therefore, it creates two VRF tables, one for each VPN. The CE routers announce IPv4 routes. The PE router installs these routes into two different VRF tables, one for each VPN. Similarly, Router PE2 creates two VRF tables into which routes are installed from the two directly connected CE routers. Router PE3 creates one VRF table because it is directly connected to only one VPN.

**Figure 6: Distribution of Routes from CE Routers to PE Routers**



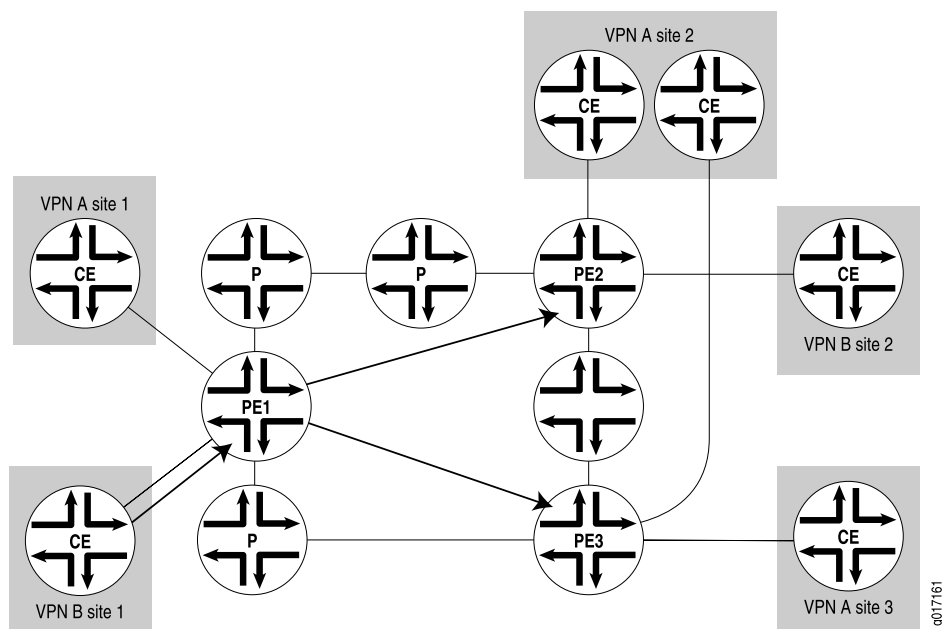
### Distribution of Routes Between PE Routers

When one PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN. If it matches, the route is converted to VPN-IPv4 format—that is, the 8-byte route distinguisher is prepended to the 4-byte VPN prefix to form a 12-byte VPN-IPv4 address. The route is then tagged with a route target community. The PE router announces the route in VPN-IPv4 format to the remote PE routers for use by VRF import policies. The routes are distributed using IBGP sessions, which are configured in the provider's core network. If the route does not match, it is not exported to other PE routers, but can still be used locally for routing, for example, if two CE routers in the same VPN are directly connected to the same PE router.

The remote PE router places the route into its `bgp.l3vpn.0` table if the route passes the import policy on the IBGP session between the PE routers. At the same time, it checks the route against the VRF import policy for the VPN. If it matches, the route distinguisher is removed from the route, and it is placed into the VRF table (the `routing-instance-name.inet.0` table) in IPv4 format.

Figure 7 on page 14 illustrates how Router PE1 distributes routes to the other PE routers in the provider's core network. Router PE2 and Router PE3 each have VRF import policies that they use to determine whether to accept routes received over the IBGP sessions and install them in their VRF tables.

Figure 7: Distribution of Routes Between PE Routers



When a PE router receives routes advertised from a directly connected CE router (Router PE1 in Figure 7 on page 14), it uses the following procedure to examine the route, convert it to a VPN route, and distribute it to the remote PE routers:

1. The PE router checks the received route using the VRF export policy for that VPN.
2. If the received route matches the export policy, the route is processed as follows:
  - a. The route is converted to VPN-IPv4 format—that is, the 8-byte route distinguisher is prepended to the 4-byte VPN prefix to form the 12-byte VPN-IPv4 address.
  - b. A route target community is added to the route.
  - c. The PE router advertises the route in VPN-IPv4 format to the remote PE routers. The routes are distributed using IBGP sessions, which are configured in the provider's core network.
3. If the route does not match the export policy, it is not exported to the remote PE routers, but can still be used locally for routing—for example, if two CE routers in the same VPN are directly connected to the same PE router.

When the remote PE router receives routes advertised from another PE router (Routers PE2 and PE3 in [Figure 7 on page 14](#)), it uses the following procedure to process the route:

1. If the route is accepted by the import policy on the IBGP session between the PE routers, the remote PE router places the route into its `bgp.l3vpn.0` table.
2. The remote PE router checks the route's route target community against the VRF import policy for the VPN.
3. If it matches, the route distinguisher is removed from the route, and it is placed into the VRF table (the `routing-instance-name.inet.0` table) in IPv4 format.

### Distribution of Routes from PE to CE Routers

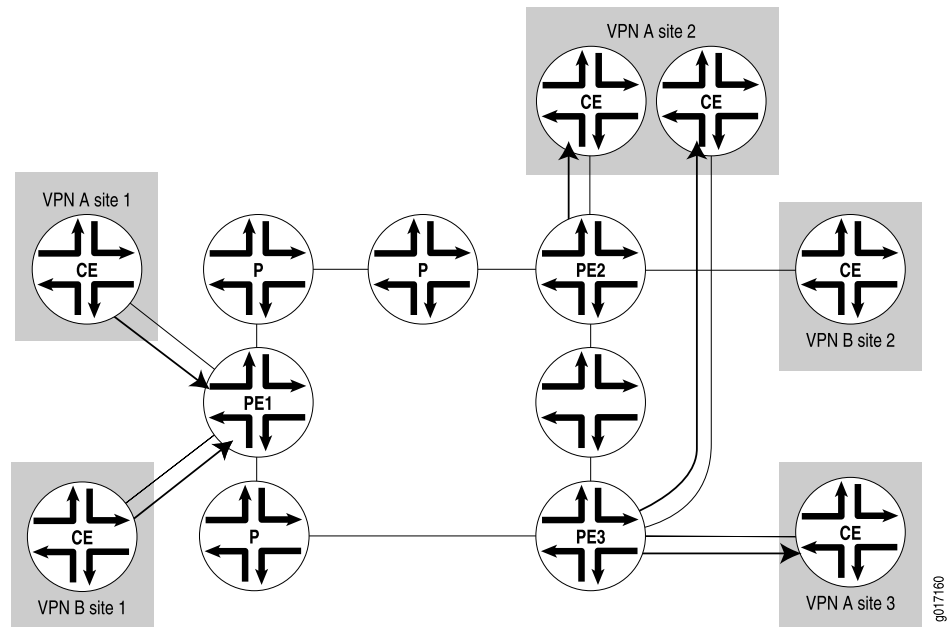
The remote PE router announces the routes in its VRF tables, which are in IPv4 format, to its directly connected CE routers.

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

- OSPF
- RIP
- BGP
- Static route

[Figure 8 on page 15](#) illustrates how the three PE routers announce their routes to their connected CE routers.

**Figure 8: Distribution of Routes from PE Routers to CE Routers**



## Forwarding Across the Provider's Core Network

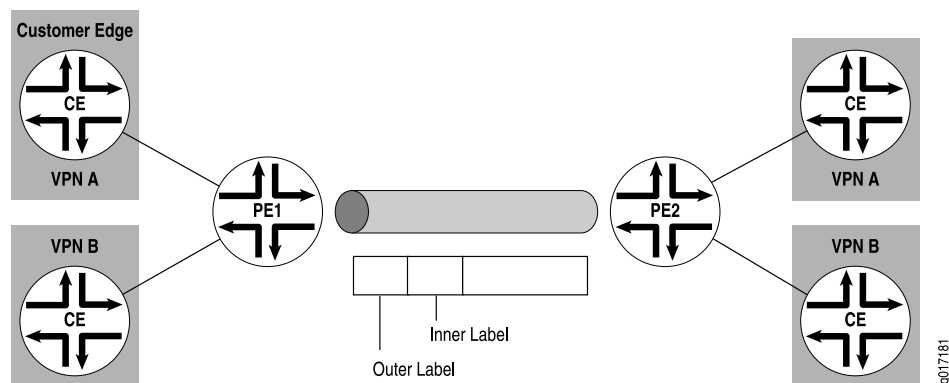
The PE routers in the provider's core network are the only routers that are configured to support VPNs and hence are the only routers to have information about the VPNs. From the point of view of VPN functionality, the provider (P) routers in the core—those P routers that are not directly connected to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

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

When PE-router-to-PE router forwarding is tunneled over MPLS label-switched paths (LSPs), the MPLS packets have a two-level label stack (see [Figure 9 on page 16](#)):

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

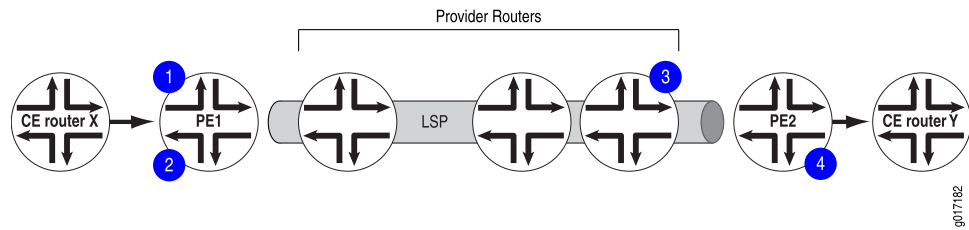
**Figure 9: Using MPLS LSPs to Tunnel Between PE Routers**



[Figure 10 on page 17](#) illustrates how the labels are assigned and removed:

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

Figure 10: Label Stack



## Routing Instances for VPNs

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

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

## Multicast over Layer 3 VPNs

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

- [Multicast over Layer 3 VPNs Overview on page 17](#)
- [Sending PIM Hello Messages to the PE Routers on page 19](#)
- [Sending PIM Join Messages to the PE Routers on page 20](#)
- [Receiving the Multicast Transmission on page 20](#)

## Multicast over Layer 3 VPNs Overview

In the unicast environment for Layer 3 VPNs, all VPN state information is contained within the PE routers. However, with multicast for Layer 3 VPNs, Protocol Independent Multicast (PIM) adjacencies are established in one of the following ways:

- You can set PIM adjacencies between the CE router and the PE router through a VRF instance at the `[edit routing-instances instance-name protocols pim]` hierarchy level. You must include the `group-address` statement for the provider tunnel, specifying a

multicast group. The rendezvous point (RP) listed within the VRF-instance is the VPN customer RP (C-RP).

- You can also set the master PIM instance and the PE's IGP neighbors by configuring statements at the [edit protocols pim] hierarchy level. You must add the multicast group specified in the VRF instance to the master PIM instance. The set of master PIM adjacencies throughout the service provider network makes up the forwarding path that becomes an RP tree rooted at the service provider RP (SP-RP). Therefore, P routers within the provider core must maintain multicast state information for the VPNs.

For this to work properly, you need two types of RP routers for each VPN:

- A C-RP—An RP router located somewhere within the VPN (can be either a service provider router or a customer router).
- An SP-RP—An RP router located within the service provider network.



**NOTE:** A PE router can act as the SP-RP and the C-RP. Moving these multicast configuration tasks to service provider routers helps to simplify the multicast Layer 3 VPN configuration process for customers. However, configuration of both SP-RP and VPN C-RP on the same PE router is not supported.

To configure multicast over a Layer 3 VPN, you must install a Tunnel Services Physical Interface Card (PIC) on the following devices:

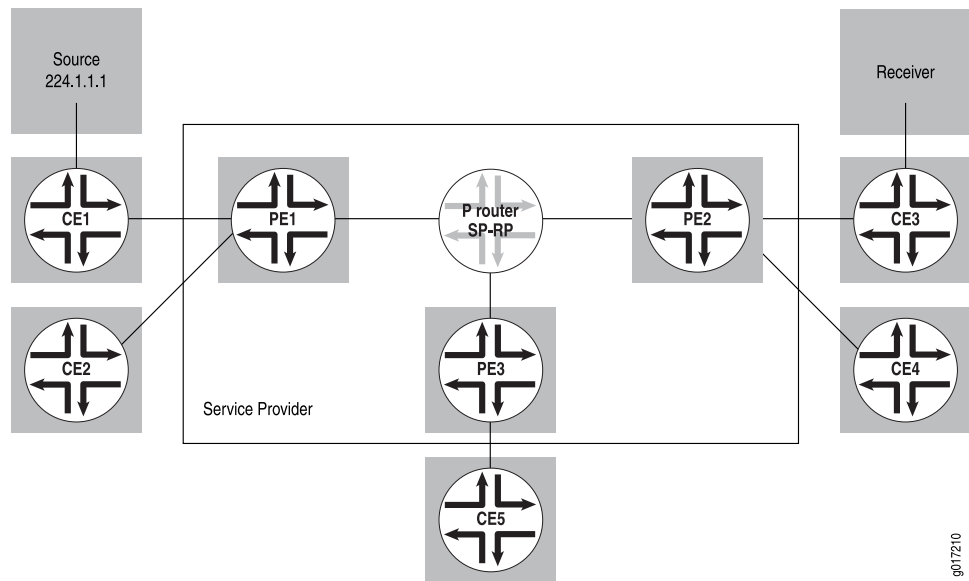
- P routers acting as RPs
- PE routers configured to run multicast routing
- CE routers acting as designated routers or as VPN-RPs

For more information about running multicast over Layer 3 VPNs, see the following documents:

- Internet draft draft-rosen-vpn-mcast-02.txt, *Multicast in MPLS/BGP VPNs*
- *Multicast Protocols Feature Guide for Routing Devices*

The sections that follow describe the operation of a multicast VPN. [Figure 11 on page 19](#) illustrates the network topology used.

Figure 11: Multicast Topology Overview



### Sending PIM Hello Messages to the PE Routers

The first step in initializing multicast over a Layer 3 VPN is the distribution of a PIM Hello message from a PE router (called PE3 in this section) to all the other PE routers on which PIM is configured.

You configure PIM on the Layer 3 VPN routing instance on the PE3 router. If a Tunnel Services PIC is installed in the routing platform, a multicast interface is created. This interface is used to communicate between the PIM instance within the VRF routing instance and the master PIM instance.

The following occurs when a PIM Hello message is sent to the PE routers:

1. A PIM Hello message is sent from the VRF routing instance over the multicast interface. A generic routing encapsulation (GRE) header is prepended to the PIM Hello message. The header message includes the VPN group address and the loopback address of the PE3 router.
2. A PIM register header is prepended to the Hello message as the packet is looped through the PIM encapsulation interface. This header contains the destination address of the SP-RP and the loopback address of the PE3 router.
3. The packet is sent to the SP-RP.
4. The SP-RP removes the top header from the packet and sends the remaining GRE-encapsulated Hello message to all the PE routers.
5. The master PIM instance on each PE router handles the GRE encapsulated packet. Because the VPN group address is contained in the packet, the master instance removes the GRE header from the packet and sends the Hello message, which contains the proper VPN group address within the VRF routing instance, over the multicast interface.

## Sending PIM Join Messages to the PE Routers

To receive a multicast broadcast from a multicast network, a CE router must send a PIM Join message to the C-RP. The process described in this section refers to [Figure 11 on page 19](#).

The CE5 router needs to receive a multicast broadcast from multicast source 224.1.1.1. To receive the broadcast, it sends a PIM Join message to the C-RP (the PE3 router):

1. The PIM Join message is sent through the multicast interface, and a GRE header is prepended to the message. The GRE header contains the VPN group ID and the loopback address of the PE3 router.
2. The PIM Join message is then sent through the PIM encapsulation interface and a register header is prepended to the packet. The register header contains the IP address of the SP-RP and the loopback address of the PE3 router.
3. The PIM Join message is sent to the SP-RP by means of unicast routing.
4. On the SP-RP, the register header is stripped off (the GRE header remains) and the packet is sent to all the PE routers.
5. The PE2 router receives the packet, and because the link to the C-RP is through the PE2 router, it sends the packet through the multicast interface to remove the GRE header.
6. Finally, the PIM Join message is sent to the C-RP.

## Receiving the Multicast Transmission

The steps that follow outline how a multicast transmission is propagated across the network:

1. The multicast source connected to the CE1 router sends the packet to group 224.1.1.1 (the VPN group address). The packet is encapsulated into a PIM register.
2. Because this packet already includes the PIM header, it is forwarded by means of unicast routing to the C-RP over the Layer 3 VPN.
3. The C-RP removes the packet and sends it out the downstream interfaces (which include the interface back to the CE3 router). The CE3 router also forwards this to the PE3 router.
4. The packet is sent through the multicast interface on the PE2 router; in the process, the GRE header is prepended to the packet.
5. Next, the packet is sent through the PIM encapsulation interface, where the register header is prepended to the data packet.
6. The packet is then forwarded to the SP-RP, which removes the register header, leaves the GRE header intact, and sends the packet to the PE routers.
7. PE routers remove the GRE header and forward the packet to the CE routers that requested the multicast broadcast by sending the PIM Join message.



**NOTE:** PE routers that have not received requests for multicast broadcasts from their connected CE routers still receive packets for the broadcast. These PE routers drop the packets as they are received.

## VPN Per-Packet Load Balancing

By default, when there are multiple equal-cost paths to the same destination for the active route, the Junos OS software uses a hash algorithm to select one of the next-hop addresses to install in the forwarding table. Whenever the set of next hops for a destination changes, this selection process (using the same hash algorithm) is repeated to choose the best single next-hop address using the same hash algorithm.

Alternatively, you can configure the Junos OS software to spread the VPN traffic across the multiple valid paths between PE devices. This feature is called per-packet load balancing. VPN traffic load balancing is only possible when more than one valid path is available. You can configure Junos OS so that, for the active route, all next-hop addresses for a destination are installed in the forwarding table. In addition to increasing the volume of traffic you can send between VPN devices, you can configure per-packet load balancing to optimize traffic flows across multiple paths.

Traffic is distributed across multiple valid paths by running a hash algorithm on various elements of the route, such as the MPLS label or the destination address. The following tables describe how the load balancing hash algorithm is run on routes at the ingress router and at the transit and egress routers. The route elements used by the hash algorithm vary depending on VPN application. If Junos OS encounters an S-bit set to 1 (indicating the bottom of the stack), it does not apply the hash algorithm any further.

**Table 3: Ingress Router Hashing**

Application	Ingress Logical Interface	MPLS Labels	Source and Destination MAC Addresses	Reordering and Flow Separation Risk	Disable Control Word	IP (Source/Destination Address and Port, Protocol)
Layer 2 VPNs and Layer 2 Circuits configured with CCC	Yes	Yes	No	Yes (if the data is variable, for example ATM)	Yes	N/A
Layer 2 VPNs and Layer 2 Circuits configured with TCC	Yes	Yes	No	Yes (if the data is variable, for example ATM)	Yes	N/A
Layer 3 VPNs and IPv4 or IPv6 RIBs	Yes	No	No	No	No	Yes
VPLS	Yes	No	Yes	No	No	Yes

Table 4: Transit and Egress Router Hashing

Application	Ingress Logical Interface	MPLS Labels (up to 3 and the S-bit is set to 1)	Reordering and Flow Separation Risk	IP (Source/Destination Address and Port, Protocol)
Layer 2 VPNs and Layer 2 Circuits configured with CCC	Yes	Yes	No	No
Layer 2 VPNs and Layer 2 Circuits configured with TCC	Yes	Yes	No	Yes
Layer 3 VPNs and IPv4 or IPv6 RIBs	Yes	Yes	No	Yes
VPLS	Yes	Yes for known unicast traffic  No for broadcast, unicast unknown, and multicast traffic	No	No

**Related Documentation** • [Configuring Protocol-Independent Load Balancing in Layer 3 VPNs on page 95](#)

## CHAPTER 2

# Introduction to Configuring Layer 3 VPNs

- [Configuring a VPN Tunnel for VRF Table Lookup on page 23](#)

## Configuring a VPN Tunnel for VRF Table Lookup

---

You can configure a VPN tunnel to facilitate VRF table lookup based on MPLS labels. You might want to enable this functionality to forward traffic on a PE-router-to-CE-device interface in a shared medium, where the CE device is a Layer 2 switch without IP capabilities (for example, a metro Ethernet switch), or to perform egress filtering at the egress PE router.

### Related Documentation

- *Example: Configuring Draft Rosen Interoperability and a VPN Tunnel Source*
- *Specifying the VT Interfaces Used by VPLS Routing Instances*



## PART 2

# Configuration

- [Configuring Layer 3 VPNs on page 27](#)
- [Layer 3 VPN Configuration Examples on page 117](#)
- [Layer 3 VPN Internet Access Examples on page 369](#)
- [Layer 3 VPN Additional Examples on page 405](#)
- [Summary of Layer 3 VPN Configuration Statements on page 513](#)



## CHAPTER 3

# Configuring Layer 3 VPNs

- [Introduction to Configuring Layer 3 VPNs on page 28](#)
- [Configuring Routing Between PE and CE Routers in Layer 3 VPNs on page 31](#)
- [Example: Configuring OSPFv2 Sham Links on page 35](#)
- [Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs on page 44](#)
- [Configuring Layer 3 VPNs to Carry IPv6 Traffic on page 46](#)
- [Configuring EBGp Multihop Sessions Between PE and CE Routers in Layer 3 VPNs on page 49](#)
- [Configuring Layer 3 VPNs to Carry IBGP Traffic on page 49](#)
- [Filtering Packets in Layer 3 VPNs Based on IP Headers on page 51](#)
- [Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs on page 58](#)
- [Load Balancing and IP Header Filtering for Layer 3 VPNs on page 59](#)
- [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 59](#)
- [Understanding Provider Edge Link Protection for BGP Labeled Unicast Paths on page 76](#)
- [Egress Protection for BGP Labeled Unicast on page 77](#)
- [Configuring Egress Protection for BGP Labeled Unicast on page 79](#)
- [Configuring a Label Allocation and Substitution Policy for VPNs on page 81](#)
- [Configuring Logical Units on the Loopback Interface for Routing Instances in Layer 3 VPNs on page 82](#)
- [Configuring Multicast Layer 3 VPNs on page 84](#)
- [Configuring Packet Forwarding for Layer 3 VPNs on page 85](#)
- [Configuring GRE Tunnels for Layer 3 VPNs on page 86](#)
- [Configuring an ES Tunnel Interface for Layer 3 VPNs on page 90](#)
- [Configuring IPsec Tunnels Instead of MPLS LSPs Between PE Routers in Layer 3 VPNs on page 92](#)
- [Configuring Protocol-Independent Load Balancing in Layer 3 VPNs on page 95](#)
- [Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes on page 97](#)
- [Configuring Traffic Policing in Layer 3 VPNs on page 98](#)

- [Configuring BGP PIC Edge for MPLS Layer 3 VPNs on page 99](#)
- [Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs on page 101](#)
- [Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112](#)

## Introduction to Configuring Layer 3 VPNs

---

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

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

```
description text;  
instance-type vrf;  
interface interface-name;  
protocols {  
    bgp {  
        group group-name {  
            peer-as as-number;  
            neighbor ip-address;  
        }  
        multihop ttl-value;  
    }  
    (ospf | ospf3) {  
        area area {  
            interface interface-name;  
        }  
        domain-id domain-id;  
        domain-vpn-tag number;  
        sham-link {  
            local address;  
        }  
        sham-link-remote address <metric number>;  
    }  
    rip {  
        rip-configuration;  
    }  
}  
route-distinguisher (as-number:id | ip-address:id);  
router-id address;  
routing-options {  
    autonomous-system autonomous-system {  
        independent-domain;  
        loops number;  
    }  
    forwarding-table {  
        export [ policy-names ];  
    }  
    interface-routes {  
        rib-group group-name;  
    }  
    martians {  
        destination-prefix match-type <allow>;
```

```

}
maximum-paths {
    path-limit;
    log-interval interval;
    log-only;
    threshold percentage;
}
maximum-prefixes {
    prefix-limit;
    log-interval interval;
    log-only;
    threshold percentage;
}
multipath {
    vpn-unequal-cost;
}
options {
    syslog (level level | upto level);
}
rib routing-table-name {
    martians {
        destination-prefix match-type <allow>;
    }
    multipath {
        vpn-unequal-cost;
    }
    static {
        defaults {
            static-options;
        }
        route destination-prefix {
            next-hop [next-hops];
            static-options;
        }
    }
}
}
static {
    defaults {
        static-options;
    }
    route destination-prefix {
        policy [ policy-names ];
        static-options;
    }
}
}
vrf-advertise-selective {
    family {
        inet-mvpn;
        inet6-mvpn;
    }
}
}
vrf-export [ policy-names ];
vrf-import [ policy-names ];
vrf-target (community | export community-name | import community-name);
vrf-table-label;

```

You can include these statements at the following hierarchy levels:

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

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

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

By default, Layer 3 VPNs are disabled.

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

**Related  
Documentation**

- [Centralized Internet Access on page 394](#)
- [Configuring Hub-and-Spoke VPN Topologies: One Interface on page 132](#)
- [Configuring Hub-and-Spoke VPN Topologies: Two Interfaces on page 144](#)
- [Configuring Overlapping VPNs Using Automatic Route Export on page 194](#)
- [Configuring Overlapping VPNs Using Routing Table Groups on page 183](#)
- [Configuring a Full-Mesh VPN Topology with Route Reflectors on page 132](#)
- [Configuring a GRE Tunnel Interface Between PE Routers on page 198](#)
- [Configuring a GRE Tunnel Interface Between a PE and CE Router on page 204](#)
- [Configuring a Simple Full-Mesh VPN Topology on page 118](#)
- [Configuring an Application-Based Layer 3 VPN Topology on page 173](#)
- [Configuring an ES Tunnel Interface Between a PE and CE Router on page 208](#)
- [Configuring an LDP-over-RSVP VPN Topology on page 159](#)
- [Configuring an OSPF Domain ID for a Layer 3 VPN on page 178](#)
- [Distributed Internet Access on page 370](#)
- [Routing Internet Traffic Through a Separate NAT Device on page 386](#)
- [Routing VPN and Internet Traffic Through Different Interfaces on page 371](#)
- [Routing VPN and Internet Traffic Through the Same Interface Bidirectionally \(VPN Has Private Addresses\) on page 382](#)
- [Routing VPN and Internet Traffic Through the Same Interface Bidirectionally \(VPN Has Public Addresses\) on page 378](#)
- [Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface on page 377](#)
- [Setting the Forwarding Class of the Ping Packets](#)

## Configuring Routing Between PE and CE Routers in Layer 3 VPNs

For the PE router to distribute VPN-related routes to and from connected CE routers, you must configure routing within the VPN routing instance. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing. For the connection to each CE router, you can configure only one type of routing.

The following sections explain how to configure VPN routing between the PE and CE routers:

- [Configuring BGP Between the PE and CE Routers on page 31](#)
- [Configuring OSPF Between the PE and CE Routers on page 32](#)
- [Configuring RIP Between the PE and CE Routers on page 33](#)
- [Configuring Static Routes Between the PE and CE Routers on page 34](#)

### Configuring BGP Between the PE and CE Routers

To configure BGP as the routing protocol between the PE and the CE routers, include the **bgp** statement:

```
bgp {
  group group-name {
    peer-as as-number;
    neighbor ip-address;
  }
}
```

You can include this statement at the following hierarchy levels:

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

Please be aware of the following limitations regarding configuring BGP for routing instances:

- In a VRF routing instance, do not configure the local autonomous system (AS) number using an AS number that is already in use by a remote BGP peer in a separate VRF routing instance. Doing so creates an autonomous system loop where all the routes received from this remote BGP peer are hidden.

You configure the local AS number using either the **autonomous-system** statement at the **[edit routing-instances *routing-instance-name* routing-options]** hierarchy level or the **local-as** statement at any of the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols bgp]**
- **[edit routing-instances *routing-instance-name* protocols bgp group *group-name*]**
- **[edit routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*]**

You configure the AS number for a BGP peer using the **peer-as** statement at the **[edit routing-instances *routing-instance-name* protocols bgp group *group-name*]** hierarchy level.

## Configuring OSPF Between the PE and CE Routers

You can configure OSPF (version 2 or version 3) to distribute VPN-related routes between PE and CE routers.

The following sections describe how to configure OSPF as a routing protocol between the PE and the CE routers:

- [Configuring OSPF Version 2 Between the PE and CE Routers on page 32](#)
- [Configuring OSPF Version 3 Between the PE and CE Routers on page 32](#)

---

### Configuring OSPF Version 2 Between the PE and CE Routers

To configure OSPF version 2 as the routing protocol between a PE and CE router, include the **ospf** statement:

```
ospf {  
  area area {  
    interface interface-name;  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

---

### Configuring OSPF Version 3 Between the PE and CE Routers

To configure OSPF version 3 as the routing protocol between a PE and CE router, include the **ospf3** statement:

```
ospf3 {  
  area area {  
    interface interface-name;  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

## Configuring RIP Between the PE and CE Routers

For a Layer 3 VPN, you can configure RIP on the PE router to learn the routes of the CE router or to propagate the routes of the PE router to the CE router. RIP routes learned from neighbors configured at any **[edit routing-instances]** hierarchy level are added to the routing instance's **inet** table (*instance\_name.inet.0*).

To configure RIP as the routing protocol between the PE and the CE router, include the **rip** statement:

```
rip {
  group group-name {
    export policy-names;
    neighbor interface-name;
  }
}
```

You can include this statement at the following hierarchy levels:

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

By default, RIP does not advertise the routes it receives. To advertise routes from a PE router to a CE router, you need to configure an export policy on the PE router for RIP. For information about how to define policies for RIP, see *Example: Applying Policies to RIP Routes Imported from Neighbors*.

To specify an export policy for RIP, include the **export** statement:

```
export [ policy-names ];
```

You can include this statement for RIP at the following hierarchy levels:

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

To install routes learned from a RIP routing instance into multiple routing tables, include the **rib-group** and **group** statements:

```
rib-group inet group-name;
group group-name {
  neighbor interface-name;
}
```

You can include these statements at the following hierarchy levels:

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

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

To configure a routing table group, include the **rib-groups** statement:

```
rib-groups group-name;
```

You can include this statement at the following hierarchy levels:

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

To add a routing table to a routing table group, include the **import-rib** statement. The first routing table name specified under the **import-rib** statement must be the name of the routing table you are configuring. For more information about how to configure routing tables and routing table groups, see the *Junos OS Routing Protocols Library for Routing Devices*.

```
import-rib [ group-names ];
```

You can include this statement at the following hierarchy levels:

- [edit routing-options rib-groups *group-name*]
- [edit logical-systems *logical-system-name* routing-options rib-groups *group-name*]

RIP instances are supported only for VRF instance types. You can configure multiple instances of RIP for VPN support only. You can use RIP in the customer edge-provider edge (CE-PE) environment to learn routes from the CE router and to propagate the PE router's instance routes in the CE router.

RIP routes learned from neighbors configured under any instance hierarchy are added to the instance's routing table, *instance-name.inet.0*.

RIP does not support routing table groups; therefore, it cannot import routes into multiple tables as the OSPF or OSPFv3 protocol does.

## Configuring Static Routes Between the PE and CE Routers

You can configure static (nonchanging) routes between the PE and CE routers of a VPN routing instance. To configure a static route for a VPN, you need to configure it within the VPN routing instance configuration at the [edit routing-instances *routing-instance-name* routing-options] hierarchy level.

To configure a static route between the PE and the CE routers, include the **static** statement:

```
static {  
  route destination-prefix {  
    next-hop [ next-hops ];  
    static-options;  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

For more information about configuring routing protocols and static routes, see the *Junos OS Routing Protocols Library for Routing Devices*.

## Example: Configuring OSPFv2 Sham Links

- [OSPFv2 Sham Links Overview on page 35](#)
- [Example: Configuring OSPFv2 Sham Links on page 36](#)

### OSPFv2 Sham Links Overview

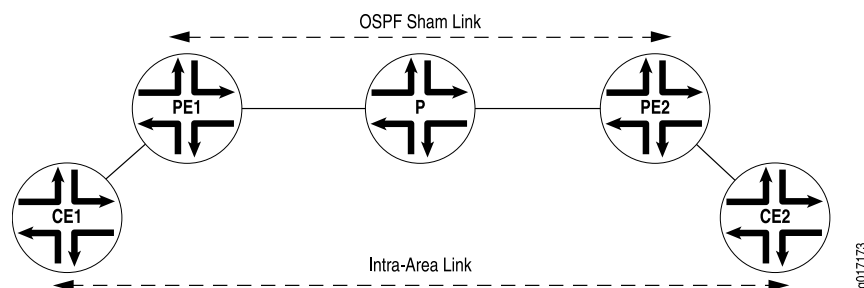
You can create an intra-area link or sham link between two provider edge (PE) routing devices so that the VPN backbone is preferred over the back-door link. A back-door link is a backup link that connects customer edge (CE) devices in case the VPN backbone is unavailable. When such a backup link is available and the CE devices are in the same OSPF area, the default behavior is to prefer this backup link over the VPN backbone. This is because the backup link is considered an intra-area link, while the VPN backbone is always considered an interarea link. Intra-area links are always preferred over interarea links.

The sham link is an unnumbered point-to-point intra-area link between PE devices. When the VPN backbone has a sham intra-area link, this sham link can be preferred over the backup link if the sham link has a lower OSPF metric than the backup link.

The sham link is advertised using Type 1 link-state advertisements (LSAs). Sham links are valid only for routing instances and OSPFv2.

Each sham link is identified by the combination of a local endpoint address and a remote endpoint address. [Figure 12 on page 35](#) shows an OSPFv2 sham link. Router CE1 and Router CE2 are located in the same OSPFv2 area. These customer edge (CE) routing devices are linked together by a Layer 3 VPN over Router PE1 and Router PE2. In addition, Router CE1 and Router CE2 are connected by an intra-area link used as a backup.

Figure 12: OSPFv2 Sham Link



OSPFv2 treats the link through the Layer 3 VPN as an interarea link. By default, OSPFv2 prefers intra-area links to interarea links, so OSPFv2 selects the backup intra-area link

as the active path. This is not acceptable in a configuration where the intra-area link is not the expected primary path for traffic between the CE routing devices. You can configure the metric for the sham link to ensure that the path over the Layer 3 VPN is preferred to a backup path over an intra-area link connecting the CE routing devices.

For the remote endpoint, you can configure the OSPFv2 interface as a demand circuit, configure IPsec authentication (you configure the actual IPsec authentication separately), and define the metric value.

You should configure an OSPFv2 sham link under the following circumstances:

- Two CE routing devices are linked together by a Layer 3 VPN.
- These CE routing devices are in the same OSPFv2 area.
- An intra-area link is configured between the two CE routing devices.

If there is no intra-area link between the CE routing devices, you do not need to configure an OSPFv2 sham link.



.....  
**NOTE:** In Junos OS Release 9.6 and later, an OSPFv2 sham link is installed in the routing table as a hidden route. Additionally, a BGP route is not exported to OSPFv2 if a corresponding OSPF sham link is available.  
.....

## Example: Configuring OSPFv2 Sham Links

This example shows how to enable OSPFv2 sham links on a PE routing device.

- [Requirements on page 36](#)
- [Overview on page 36](#)
- [Configuration on page 37](#)
- [Verification on page 43](#)

### Requirements

---

No special configuration beyond device initialization is required before configuring this example.

### Overview

---

The sham link is an unnumbered point-to-point intra-area link and is advertised by means of a type 1 link-state advertisement (LSA). Sham links are valid only for routing instances and OSPFv2.

Each sham link is identified by a combination of the local endpoint address and a remote endpoint address and the OSPFv2 area to which it belongs. You manually configure the sham link between two PE devices, both of which are within the same VPN routing and forwarding (VRF) routing instance, and you specify the address for the local end point of the sham link. This address is used as the source for the sham link packets and is also used by the remote PE routing device as the sham link remote end point. You can also include the optional **metric** option to set a metric value for the remote end point. The

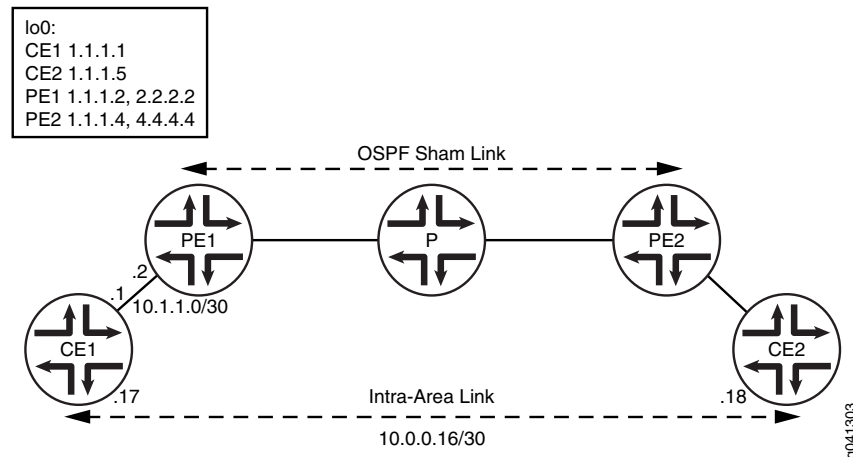
metric value specifies the cost of using the link. Routes with lower total path metrics are preferred over those with higher path metrics.

To enable OSPFv2 sham links on a PE routing device:

- Configure an extra loopback interface on the PE routing device.
- Configure the VRF routing instance that supports Layer 3 VPNs on the PE routing device, and associate the sham link with an existing OSPF area. The OSPFv2 sham link configuration is also included in the routing instance. You configure the sham link's local endpoint address, which is the loopback address of the local VPN, and the remote endpoint address, which is the loopback address of the remote VPN. In this example, the VRF routing instance is named red.

Figure 13 on page 37 shows an OSPFv2 sham link.

Figure 13: OSPFv2 Sham Link Example



The devices in the figure represent the following functions:

- CE1 and CE2 are the customer edge devices.
- PE1 and PE2 are the provider edge devices.
- P is the provider device.

“CLI Quick Configuration” on page 37 shows the configuration for all of the devices in Figure 13 on page 37. The section “Step-by-Step Procedure” on page 39 describes the steps on Device PE1.

### Configuration

#### CLI Quick Configuration

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

```
CE1 set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.1/30
    set interfaces fe-1/2/0 unit 0 family mpls
```

```

set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.17/30
set interfaces lo0 unit 0 family inet address 1.1.1.1/32
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0 metric 100
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 1

PE1  set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.2/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.5/30
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.2/32
set interfaces lo0 unit 1 family inet address 2.2.2.2/32
set protocols mpls interface fe-1/2/1.0
set protocols bgp group toR4 type internal
set protocols bgp group toR4 local-address 1.1.1.2
set protocols bgp group toR4 family inet-vpn unicast
set protocols bgp group toR4 neighbor 1.1.1.4
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface fe-1/2/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement bgp-to-ospf term 1 from protocol bgp
set policy-options policy-statement bgp-to-ospf term 1 then accept
set policy-options policy-statement bgp-to-ospf term 2 then reject
set routing-instances red instance-type vrf
set routing-instances red interface fe-1/2/0.0
set routing-instances red interface lo0.1
set routing-instances red route-distinguisher 2:1
set routing-instances red vrf-target target:2:1
set routing-instances red protocols ospf export bgp-to-ospf
set routing-instances red protocols ospf sham-link local 2.2.2.2
set routing-instances red protocols ospf area 0.0.0.0 sham-link-remote 4.4.4.4 metric
    10
set routing-instances red protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set routing-instances red protocols ospf area 0.0.0.0 interface lo0.1
set routing-options router-id 1.1.1.2
set routing-options autonomous-system 2

P    set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.6/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.9/30
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 3 family inet address 1.1.1.3/32
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface all
set protocols ldp interface all
set routing-options router-id 1.1.1.3

PE2  set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.10/30
set interfaces fe-1/2/0 unit 0 family mpls

```

```

set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.13/30
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.4/32
set interfaces lo0 unit 1 family inet address 4.4.4.4/32
set protocols mpls interface fe-1/2/0.0
set protocols bgp group toR2 type internal
set protocols bgp group toR2 local-address 1.1.1.4
set protocols bgp group toR2 family inet-vpn unicast
set protocols bgp group toR2 neighbor 1.1.1.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ldp interface fe-1/2/0.0
set protocols ldp interface lo0.0
set policy-options policy-statement bgp-to-ospf term 1 from protocol bgp
set policy-options policy-statement bgp-to-ospf term 1 then accept
set policy-options policy-statement bgp-to-ospf term 2 then reject
set routing-instances red instance-type vrf
set routing-instances red interface fe-1/2/1.0
set routing-instances red interface lo0.1
set routing-instances red route-distinguisher 2:1
set routing-instances red vrf-target target:2:1
set routing-instances red protocols ospf export bgp-to-ospf
set routing-instances red protocols ospf sham-link local 4.4.4.4
set routing-instances red protocols ospf area 0.0.0.0 sham-link-remote 2.2.2.2 metric 10
set routing-instances red protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set routing-instances red protocols ospf area 0.0.0.0 interface lo0.1
set routing-options router-id 1.1.1.4
set routing-options autonomous-system 2

```

```

CE2 set interfaces fe-1/2/0 unit 14 family inet address 10.1.1.14/30
set interfaces fe-1/2/0 unit 14 family mpls
set interfaces fe-1/2/0 unit 18 family inet address 10.0.0.18/30
set interfaces lo0 unit 5 family inet address 1.1.1.5/32
set protocols ospf area 0.0.0.0 interface fe-1/2/0.14
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.18
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 1.1.1.5
set routing-options autonomous-system 3

```

**Step-by-Step Procedure** The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Modifying the Junos OS Configuration* in *CLI User Guide*.

To configure OSPFv2 sham links on each PE device:

1. Configure the interfaces, including two loopback interfaces.

```

[edit interfaces]
user@PE1# set fe-1/2/0 unit 0 family inet address 10.1.1.2/30
user@PE1# set fe-1/2/0 unit 0 family mpls
user@PE1# set fe-1/2/1 unit 0 family inet address 10.1.1.5/30
user@PE1# set fe-1/2/1 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 1.1.1.2/32

```

```
user@PE1# set lo0 unit 1 family inet address 2.2.2.2/32
```

2. Configure MPLS on the core-facing interface.

```
[edit protocols mpls]
user@PE1# set interface fe-1/2/1.0
```

3. Configure internal BGP (IBGP).

```
[edit ]
user@PE1# set protocols bgp group toR4 type internal
user@PE1# set protocols bgp group toR4 local-address 1.1.1.2
user@PE1# set protocols bgp group toR4 family inet-vpn unicast
user@PE1# set protocols bgp group toR4 neighbor 1.1.1.4
```

4. Configure OSPF on the core-facing interface and on the loopback interface that is being used in the main instance.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface fe-1/2/1.0
user@PE1# set interface lo0.0 passive
```

5. Configure LDP or RSVP on the core-facing interface and on the loopback interface that is being used in the main instance.

```
[edit protocols ldp]
user@PE1# set interface fe-1/2/1.0
user@PE1# set interface lo0.0
```

6. Configure a routing policy for use in the routing instance.

```
[edit policy-options policy-statement bgp-to-ospf]
user@PE1# set term 1 from protocol bgp
user@PE1# set term 1 then accept
user@PE1# set term 2 then reject
```

7. Configure the routing instance.

```
[edit routing-instances red]
user@PE1# set instance-type vrf
user@PE1# set interface fe-1/2/0.0
user@PE1# set route-distinguisher 2:1
user@PE1# set vrf-target target:2:1
user@PE1# set protocols ospf export bgp-to-ospf
user@PE1# set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
```

8. Configure the OSPFv2 sham link.

Include the extra loopback interface in the routing instance and also in the OSPF configuration.

Notice that the metric on the sham-link interface is set to 10. On Device CE1's backup OSPF link, the metric is set to 100. This causes the sham link to be the preferred link.

```
[edit routing-instances red]
user@PE1# set interface lo0.1
user@PE1# set protocols ospf sham-link local 2.2.2.2
user@PE1# set protocols ospf area 0.0.0.0 sham-link-remote 4.4.4.4 metric 10
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.1
```

9. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options]
user@PE1# set router-id 1.1.1.2
user@PE1# set autonomous-system 2
```

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

```
[edit]
user@R1# commit
```

**Results** Confirm your configuration by entering the **show interfaces** and the **show routing-instances** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Output for PE1:

```
user@PE1# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.1.1.2/30;
    }
    family mpls;
  }
}
fe-1/2/1 {
  unit 0 {
    family inet {
      address 10.1.1.5/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 1.1.1.2/32;
    }
  }
  unit 1 {
    family inet {
      address 2.2.2.2/32;
    }
  }
}

user@PE1# show protocols
mpls {
  interface fe-1/2/1.0;
}
bgp {
  group toR4 {
    type internal;
    local-address 1.1.1.2;
    family inet-vpn {
```

```
        unicast;
    }
    neighbor 1.1.1.4;
}
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface fe-1/2/1.0;
    interface lo0.0;
}

user@PE1# show policy-options
policy-statement bgp-to-ospf {
    term 1 {
        from protocol bgp;
        then accept;
    }
    term 2 {
        then reject;
    }
}

user@PE1# show routing-instances
red {
    instance-type vrf;
    interface fe-1/2/0.0;
    interface lo0.1;
    route-distinguisher 2:1;
    vrf-target target:2:1;
    protocols {
        ospf {
            export bgp-to-ospf;
            sham-link local 2.2.2.2;
            area 0.0.0.0 {
                sham-link-remote 4.4.4.4 metric 10;
                interface fe-1/2/0.0;
                interface lo0.1;
            }
        }
    }
}

user@PE1# show routing-options
router-id 1.1.1.2;
autonomous-system 2;
```

## Verification

Confirm that the configuration is working properly.

- [Verifying the Sham Link Interfaces on page 43](#)
- [Verifying the Local and Remote End Points of the Sham Link on page 43](#)
- [Verifying the Sham Link Adjacencies on page 43](#)
- [Verifying the Link-State Advertisement on page 44](#)
- [Verifying the Path Selection on page 44](#)

### Verifying the Sham Link Interfaces

**Purpose** Verify the sham link interface. The sham link is treated as an interface in OSPFv2, with the named displayed as **shamlink.<unique identifier>**, where the unique identifier is a number. For example, **shamlink.0**. The sham link appears as a point-to-point interface.

**Action** From operational mode, enter the **show ospf interface instance *instance-name*** command.

```
user@PE1> show ospf interface instance red
Interface      State   Area      DR ID      BDR ID      Nbrs
100.1          DR      0.0.0.0    2.2.2.2    0.0.0.0      0
fe-1/2/0.0     PtToPt  0.0.0.0    0.0.0.0    0.0.0.0      1
shamlink.0     PtToPt  0.0.0.0    0.0.0.0    0.0.0.0      1
```

### Verifying the Local and Remote End Points of the Sham Link

**Purpose** Verify the local and remote end points of the sham link. The MTU for the sham link interface is always zero.

**Action** From operational mode, enter the **show ospf interface instance *instance-name* detail** command.

```
user@PE1> show ospf interface shamlink.0 instance red
Interface      State   Area      DR ID      BDR ID      Nbrs
shamlink.0     PtToPt  0.0.0.0    0.0.0.0    0.0.0.0      1

Type: P2P, Address: 0.0.0.0, Mask: 0.0.0.0, MTU: 0, Cost: 10
Local: 2.2.2.2, Remote: 4.4.4.4
Adj count: 1
Hello: 10, Dead: 40, ReXmit: 5, Not Stub
Auth type: None
Protection type: None, No eligible backup
Topology default (ID 0) -> Cost: 10
```

### Verifying the Sham Link Adjacencies

**Purpose** Verify the adjacencies between the configured sham links.

**Action** From operational mode, enter the **show ospf neighbor instance *instance-name*** command.

```
user@PE1> show ospf neighbor instance red
Address        Interface      State   ID          Pri   Dead
10.1.1.1       fe-1/2/0.0    Full    1.1.1.1     128   35
4.4.4.4        shamlink.0     Full    4.4.4.4      0     31
```

**Verifying the Link-State Advertisement**

**Purpose** Verify that the router LSA originated by the instance carries the sham link adjacency as an unnumbered point-to-point link. The link data for sham links is a number ranging from 0x80010000 through 0x8001ffff.

**Action** From operational mode, enter the **show ospf database instance** *instance-name* command.

```
user@PE1> show ospf database instance red
```

```

      OSPF database, Area 0.0.0.0
  Type      ID          Adv Rtr      Seq          Age  Opt  Cksum  Len
  Router    1.1.1.1      1.1.1.1      0x80000009   1803 0x22 0x6ec7  72
  Router    1.1.1.5      1.1.1.5      0x80000007    70 0x22 0x2746  72
  Router    *2.2.2.2      2.2.2.2      0x80000006    55 0x22 0xda6b  60
  Router    4.4.4.4      4.4.4.4      0x80000005    63 0x22 0xb19  60
  Network   10.0.0.18      1.1.1.5      0x80000002    70 0x22 0x9a71  32

      OSPF AS SCOPE link state database
  Type      ID          Adv Rtr      Seq          Age  Opt  Cksum  Len
  Extern    2.2.2.2      4.4.4.4      0x80000002    72 0xa2 0x343  36
  Extern    *4.4.4.4      2.2.2.2      0x80000002    71 0xa2 0xe263  36

```

**Verifying the Path Selection**

**Purpose** Verify that the Layer 3 VPN path is used instead of the backup path.

**Action** From operational mode, enter the **traceroute** command from Device CE1 to Device CE2.

```
user@CE1> traceroute 1.1.1.5
```

```

traceroute to 1.1.1.5 (1.1.1.5), 30 hops max, 40 byte packets
 1  10.1.1.2 (10.1.1.2)  1.930 ms  1.664 ms  1.643 ms
 2  * * *
 3  10.1.1.10 (10.1.1.10)  2.485 ms  1.435 ms  1.422 ms
    MPLS Label=299808 CoS=0 TTL=1 S=1
 4  1.1.1.5 (1.1.1.5)  1.347 ms  1.362 ms  1.329 ms

```

**Meaning** The traceroute operation shows that the Layer 3 VPN is the preferred path. If you were to remove the sham link or if you were to modify the OSPF metric to prefer that backup path, the traceroute would show that the backup path is preferred.

**Related Documentation**

- *OSPF Configuration Overview*
- *Junos OS VPNs Library for Routing Devices*

## Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs

You can configure a maximum limit on the number of prefixes and paths that can be installed into the routing tables. Using prefix and path limits, you can curtail the number of prefixes and paths received from a CE router in a VPN. Prefix and path limits apply only to dynamic routing protocols, and are not applicable to static or interface routes.

To limit the number of paths accepted by a PE router from a CE router, include the **maximum-paths** statement:

**maximum-paths** *path-limit* <log-interval *interval* | log-only | threshold *percentage*>;

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

Specify the **log-only** option to generate warning messages only (an advisory limit). Specify the **threshold** option to generate warnings before the limit is reached. Specify the **log-interval** option to configure the minimum time interval between log messages.

There are two modes for route limits: advisory and mandatory. An advisory limit triggers warnings. A mandatory limit rejects additional routes after the limit is reached.



**NOTE:** Application of a route limit may result in unpredictable dynamic routing protocol behavior. For example, when the limit is reached and routes are rejected, BGP may not reinstall the rejected routes after the number of routes drops back below the limit. BGP sessions may need to be cleared.

To limit the number of prefixes accepted by a PE router from a CE router, include the **maximum-prefixes** statement:

**maximum-prefixes** *prefix-limit* <log-interval *interval* | log-only | threshold *percentage*>;

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

There are two modes for route limits: advisory and mandatory. An advisory limit triggers warnings. A mandatory limit rejects additional routes after the limit is reached.



**NOTE:** Application of a route limit may result in unpredictable dynamic routing protocol behavior. For example, when the limit is reached and routes are rejected, BGP may not reinstall the rejected routes after the number of routes drops back below the limit. BGP sessions may need to be cleared.

A mandatory path or prefix limit, in addition to triggering a warning message, rejects any additional paths or prefixes once the limit is reached.



**NOTE:** Setting a path or prefix limit might result in unpredictable dynamic routing protocol behavior.

You can also configure the following options for both the **maximum-paths** and **maximum-prefixes** statements:

- **log-interval**—Specify the interval at which log messages are sent. This option generates warning messages only (an advisory limit).

Specify the **log-interval** option to configure the minimum time interval between log messages.

- **log-only**—Generate warning messages only. No limit is placed on the number of paths or prefixes stored in the routing tables.
- **threshold**—Generate warning messages after the specified percentage of the maximum paths or prefixes has been reached.

## Configuring Layer 3 VPNs to Carry IPv6 Traffic

---

You can configure IP version 6 (IPv6) between the PE and CE routers of a Layer 3 VPN. The PE router must have the PE router to PE router BGP session configured with the **family inet6-vpn** statement. The CE router must be capable of receiving IPv6 traffic. You can configure BGP or static routes between the PE and CE routers.

The following sections explain how to configure IPv6 VPNs between the PE routers:

- [Configuring IPv6 on the PE Router on page 46](#)
- [Configuring the Connection Between the PE and CE Routers on page 46](#)
- [Configuring IPv6 on the Interfaces on page 48](#)

### Configuring IPv6 on the PE Router

To configure IPv6 between the PE and CE routers, include the **family inet6-vpn** statement in the configuration on the PE router:

```
family inet6-vpn {  
  (any | multicast | unicast) {  
    aggregate-label community community-name;  
    prefix-limit maximum prefix-limit;  
    rib-group rib-group-name;  
  }  
}
```

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

You also must include the **ipv6-tunneling** statement:

```
ipv6-tunneling;
```

You can include this statement at the following hierarchy levels:

- **[edit protocols mpls]**
- **[edit logical-systems *logical-system-name* protocols mpls]**

### Configuring the Connection Between the PE and CE Routers

To support IPv6 routes, you must configure BGP, OSPF version 3, IS-IS, or static routes for the connection between the PE and CE routers in the Layer 3 VPN. You can configure BGP to handle just IPv6 routes or both IP version 4 (IPv4) and IPv6 routes.

For more information about IPv6, see the *Junos OS Routing Protocols Library for Routing Devices*.

For more information about IS-IS see *Example: Configuring IS-IS*,

The following sections explain how to configure BGP and static routes:

- [Configuring BGP on the PE Router to Handle IPv6 Routes on page 47](#)
- [Configuring BGP on the PE Router for IPv4 and IPv6 Routes on page 47](#)
- [Configuring OSPF Version 3 on the PE Router on page 48](#)
- [Configuring Static Routes on the PE Router on page 48](#)

### Configuring BGP on the PE Router to Handle IPv6 Routes

To configure BGP in the Layer 3 VPN routing instance to handle IPv6 routes, include the **bgp** statement:

```
bgp {
  group group-name {
    local-address IPv6-address;
    family inet6 {
      unicast;
    }
    peer-as as-number;
    neighbor IPv6-address;
  }
}
```

You can include this statement at the following hierarchy levels:

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

### Configuring BGP on the PE Router for IPv4 and IPv6 Routes

To configure BGP in the Layer 3 VPN routing instance to handle both IPv4 and IPv6 routes, include the **bgp** statement:

```
bgp {
  group group-name {
    local-address IPv4-address;
    family inet {
      unicast;
    }
    family inet6 {
      unicast;
    }
    peer-as as-number;
    neighbor address;
  }
}
```

You can include this statement at the following hierarchy levels:

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

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

---

### Configuring OSPF Version 3 on the PE Router

To configure OSPF version 3 in the Layer 3 VPN routing instance to handle IPv6 routes, include the **ospf3** statement:

```
ospf3 {  
  area area-id {  
    interface interface-name;  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

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

---

### Configuring Static Routes on the PE Router

To configure a static route to the CE router in the Layer 3 VPN routing instance, include the **routing-options** statement:

```
routing-options {  
  rib routing-table.inet6.0 {  
    static {  
      defaults {  
        static-options;  
      }  
    }  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

## Configuring IPv6 on the Interfaces

You need to configure IPv6 on the PE router interfaces to the CE routers and on the CE router interfaces to the PE routers.

To configure the interface to handle IPv6 routes, include the **family inet6** statement:

```
family inet6 {  
  address ipv6-address;  
}
```

You can include this statement at the following hierarchy levels:

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

If you have configured the Layer 3 VPN to handle both IPv4 and IPv6 routes, configure the interface to handle both IPv4 and IPv6 routes by including the **unit** statement:

```
unit unit-number {
  family inet {
    address ipv4-address;
  }
  family inet6 {
    address ipv6-address;
  }
}
```

You can include this statement at the following hierarchy levels:

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

## Configuring EBGp Multihop Sessions Between PE and CE Routers in Layer 3 VPNs

You can configure an EBGp or IBGP multihop session between the PE and CE routers of a Layer 3 VPN. This allows you to have one or more routers between the PE and CE routers. Using IBGP between PE and CE routers does not require the configuration of any additional statements. However, using EBGp between the PE and CE routers requires the configuration of the **multihop** statement.

To configure an external BGP multihop session for the connection between the PE and CE routers, include the **multihop** statement on the PE router. To help prevent routing loops, you have to configure a time-to-live (TTL) value for the multihop session:

```
multihop tll-value;
```

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

## Configuring Layer 3 VPNs to Carry IBGP Traffic

An independent AS domain is separate from the primary routing instance domain. An AS is a set of routers that are under a single technical administration and that generally use a single IGP and metrics to propagate routing information within the set of routers. An AS appears to other ASs to have a single, coherent interior routing plan and presents a consistent picture of what destinations are reachable through it.

Configuring an independent domain allows you to keep the AS paths of the independent domain from being shared with the AS path and AS path attributes of other domains, including the master routing instance domain.

If you are using BGP on the router, you must configure an AS number.

When you configure BGP as the routing protocol between a PE router and a CE router in a Layer 3 VPN, you typically configure external peering sessions between the Layer 3 VPN service provider and the customer network ASs.

If the customer network has several sites advertising routes through an external BGP session to the service provider network and if the same AS is used by all the customer sites, the CE routers reject routes from the other CE routers. They detect a loop in the BGP AS path attribute.

To prevent the CE routers from rejecting each other's routes, you could configure the following:

- PE routers advertising routes received from remote PE routers can remap the customer network AS number to its own AS number.
- AS path loops can be configured.
- The customer network can be configured with different AS numbers at each site.

These types of configurations can work when there are no BGP routing exchanges between the customer network and other networks. However, they do have limitations for customer networks that use BGP internally for purposes other than carrying traffic between the CE routers and the PE routers. When those routes are advertised outside the customer network, the service provider ASs are present in the AS path.

To improve the transparency of Layer 3 VPN services for customer networks, you can configure the routing instance for the Layer 3 VPN to isolate the customer's network attributes from the service provider's network attributes.

When you include the **independent-domain** statement in the Layer 3 VPN routing instance configuration, BGP attributes received from the customer network (from the CE router) are stored in a BGP attribute (ATTRSET) that functions like a stack. When that route is advertised from the remote PE router to the remote CE router, the original BGP attributes are restored. This is the default behavior for BGP routes that are advertised to Layer 3 VPNs located in different domains.

This functionality is described in the Internet draft *draft-marques-ppvnp-ibgp-version.txt*, *RFC 2547bis Networks Using Internal BGP as PE-CE Protocol*.

To allow a Layer 3 VPN to transport IBGP traffic, include the **independent-domain** statement:

**independent-domain;**

You can include this statement at the following hierarchy levels:

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



**NOTE:** All PE routers participating in a Layer 3 VPN with the **independent-domain** statement in its configuration must be running Junos OS Release 6.3 or later.

The independent domain uses the transitive path attribute 128 (attribute set) to tunnel the independent domain's BGP attributes through the Internal BGP (IBGP) core. In Junos OS Release 10.3 and later, if BGP receives attribute 128 and you have not configured an independent domain in any routing instance, BGP treats the received attribute 128 as an unknown attribute.

There is a limit of 16 ASs for each domain.

**Related Documentation**

- [Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains on page 358](#)
- [Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection](#)

## Filtering Packets in Layer 3 VPNs Based on IP Headers

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

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

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

- Perform egress filtering at the egress PE router.

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

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

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

```
vrf-table-label {
```

```
    source-class-usage;  
}
```

You can include the statement at the following hierarchy levels:

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

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

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

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

- [Egress Filtering Options on page 52](#)
- [Support on Aggregated and VLAN Interfaces for IP-Based Filtering on page 53](#)
- [Support on ATM and Frame Relay Interfaces for IP-Based Filtering on page 53](#)
- [Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering on page 54](#)
- [Support on SONET/SDH and DS3/E3 Channelized Enhanced Intelligent Queuing Interfaces for IP-Based Filtering on page 54](#)
- [Support on Multilink PPP and Multilink Frame Relay Interfaces for IP-Based Filtering on page 56](#)
- [Support for IP-Based Filtering of Packets with Null Top Labels on page 56](#)
- [General Limitations on IP-Based Filtering on page 57](#)

## Egress Filtering Options

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

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

## Support on Aggregated and VLAN Interfaces for IP-Based Filtering

Support for the **vrf-table-label** statement over aggregated and VLAN interfaces is available on the routers summarized in [Table 5 on page 53](#).

**Table 5: Support for Aggregated and VLAN Interfaces**

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



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

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

Support for the **vrf-table-label** statement over Asynchronous Transfer Mode (ATM) and Frame Relay interfaces is available on the routers summarized in [Table 6 on page 53](#).

**Table 6: Support for ATM and Frame Relay Interfaces**

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

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

- The **vrf-table-label** statement is supported on ATM interfaces, but with the following limitations:
  - ATM interfaces can be configured on the M320 router and the T Series routers, and on M Series routers with an enhanced FPC.
  - The interface can only be a PE router interface receiving traffic from a P router.

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

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

Support for the **vrf-table-label** statement over Ethernet, SONET/SDH, and T1/T3/E3 interfaces is available on the routers summarized in [Table 7 on page 54](#).

**Table 7: Support for Ethernet, SONET/SDH, and T1/T3/E3 Interfaces**

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

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

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

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

Support for the **vrf-table-label** statement for the specified channelized IQE interfaces is only available on M120 and M320 routers with Enhanced III FPCs as summarized in [Table 8 on page 54](#).

**Table 8: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs**

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

**Table 8: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs** (*continued*)

Interfaces	M120 Routers with Enhanced III FPCs	M320 Routers with Enhanced III FPCs
STM1	Yes	Yes
DS3	Yes	Yes
E3	Yes	Yes

The following IQE Type-1 PICs are supported:

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

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

- Multiport IQE PIC interfaces constraints—On multiport IQE PICs, such as the 2-port Channelized OC3/STM1 IQE with SFP, if the port 1 interface is configured as one logical system with its own routing-instance and the port 2 interface is configured as a different logical system with its own routing instances such that there are core-facing logical interfaces on both port 1 and port 2, then you cannot configure the **vrf-table-label** statement on routing-instance in both logical systems. Only one set of LSI labels are supported; the last routing instance with the **vrf-table-label** statement configured is committed.
- Frame Relay encapsulation and logical interfaces across logical systems constraints—Similar to the multiport PIC with logical systems, if you try to configure one logical interface of an IQE PIC with Frame Relay encapsulation in one logical system and configure another logical interface on the same IQE PIC in the second logical system, the configuration will not work for all the **vrf-table-label** statement configured instances. It will only work for the instances configured in one of the logical systems.

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

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

Support for the **vrf-table-label** statement over Multilink Point-to-Point Protocol (MLPPP) and Multilink Frame Relay (MLFR) interfaces is available on the routers summarized in [Table 9 on page 56](#).

**Table 9: Support for Multilink PPP and Multilink Frame Relay Interfaces**

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

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

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

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

- 1-port Gigabit Ethernet with SFP
- 2-port Gigabit Ethernet with SFP
- 4-port Gigabit Ethernet with SFP
- 10-port Gigabit Ethernet with SFP
- 1-port SONET STM4
- 4-port SONET STM4
- 1-port SONET STM16
- 1-port SONET STM16 (non-SFP)
- 4-port SONET STM16
- 1-port SONET STM64

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

- 1-port 10-Gigabit Ethernet

- 1-port 10-Gigabit Ethernet IQ2

## General Limitations on IP-Based Filtering

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

- Firewall filters cannot be applied to interfaces included in a routing instance on which you have configured the **vrf-table-label** statement.
- The time-to-live (TTL) value in the MPLS header is not copied back to the IP header of packets sent from the PE router to the CE router.
- You cannot include the **vrf-table-label** statement in a routing instance configuration that also includes a virtual loopback tunnel interface; the commit operation fails in this case.
- You can include the statement in the configuration for Multilink Frame Relay (MLFR FRF.16) encapsulated PE-router-to-P-router interfaces only on J Series routers.
- When you include the statement, MPLS packets with label-switched interface (LSI) labels that arrive on core-facing interfaces are not counted at the logical interface level if the core-facing interface is any of the following:
  - ATM
  - Frame Relay
  - Ethernet configured with VLANs
  - Aggregated Ethernet configured with VLANs
- You cannot include the statement in the configuration of a VRF routing instance if the PE-router-to-P-router interface is any of the following interfaces:
  - Aggregated SONET/SDH interface
  - Channelized interface
  - Tunnel interface (for example, generic routing encapsulation [GRE] or IP Security [IPsec])
  - Circuit cross-connect (CCC) or translational cross-connect (TCC) encapsulated interface
  - Logical tunnel interface
  - Virtual private LAN service (VPLS) encapsulated interface



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

- You cannot include the **vrf-table-label** statement in the configuration of a VRF routing instance if the PE-router-to-P-router PIC is one of the following PICs:
  - 10-port E1
  - 8-port Fast Ethernet

- 12-port Fast Ethernet
- 48-port Fast Ethernet
- ATM PIC other than the ATM2 IQ
- Label-switched interface (LSI) traffic statistics are not supported for Intelligent Queuing 2 (IQ2), Enhanced IQ (IQE), and Enhanced IQ2 (IQ2E) PICs on M Series routers.

**Related  
Documentation**

- *Enabling Source Class and Destination Class Usage*

---

## Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs

When you include the **vrf-table-label** statement in the configuration for a routing instance (as described in [“Filtering Packets in Layer 3 VPNs Based on IP Headers” on page 51](#)) but do not explicitly apply a classifier to the routing instance, the default MPLS EXP classifier is applied.

For PICs that are installed on Enhanced FPCs, you can apply a custom classifier to override the default MPLS EXP classifier for the routing instance. For detailed instructions, see the *Class of Service Feature Guide for Routing Devices*. The following instructions serve as a summary:

1. Filter traffic based on the IP header by including the **vrf-table-label** statement at the **[edit routing-instances routing-instance-name]** hierarchy level:

```
[edit routing-instances routing-instance-name]
vrf-table-label;
```

2. Configure a custom MPLS EXP classifier by including the appropriate statements at the **[edit class-of-service]** hierarchy level. For instructions, see the *Class of Service Feature Guide for Routing Devices*.
3. Configure the routing instance for CoS by including the **routing-instances** statement at the **[edit class-of-service]** hierarchy level:

```
[edit class-of-service]
routing-instances routing-instance-name {
  classifiers {
    exp (classifier-name | default);
  }
}
```

4. Configure the routing instance to use the custom MPLS EXP classifier by including the **classifiers** statement at the **[edit class-of-service routing-instances routing-instance-name]** hierarchy level:

```
[edit class-of-service routing-instances routing-instance-name]
classifiers {
  exp classifier-name;
}
```

To display the MPLS EXP classifiers associated with all routing instances, issue the **show class-of-service routing-instances** command.



**NOTE:** The following caveats apply to custom MPLS EXP classifiers for routing instances:

- An Enhanced FPC is required.
- Logical systems are not supported.

## Load Balancing and IP Header Filtering for Layer 3 VPNs

You can now simultaneously enable both load balancing of traffic across both internal and external BGP paths and filtering of traffic based on the IP header. This enables you to configure filters and policers at the egress PE router for traffic that is simultaneously being load-balanced across both internal and external BGP paths. This feature is available only on the M120 router, M320 router, MX Series routers, and T Series routers.

To enable these features on a Layer 3 VPN routing instance, include the **vpn-unequal-cost equal-external-internal** statement at the **[edit routing-instances *routing-instance-name* routing-options multipath]** hierarchy level and the **vrf-table-label** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level.

If you issue the **show route detail** command, you can discover whether or not a route is being load-balanced (equal-external-internal) and what its interface index is.

If you have also configured fast reroute, please be aware of the following behavior:

- If an IBGP path goes down, it could be replaced by either an active EBGp path or an active IBGP path.
- If an EBGp path goes down, it can only be replaced by another active EBGp path. This prevents the forwarding of core-facing interface traffic to an IBGP destination.



**NOTE:** You can include the **vpn-unequal-cost equal-external-internal** statement and the **l3vpn** statement at the **[edit routing-options forwarding-options chained-composite-next-hop ingress]** hierarchy level simultaneously. However, if you do this, EBGp does not work. This means that when there are both paths with chained nexthops and paths with nonchained nexthops as candidates for EBGp equal-cost multipath (ECMP), the paths using chained nexthops are excluded. In a typical case, the excluded paths are the internal paths.

## Example: Configuring Provider Edge Link Protection in Layer 3 VPNs

- [Understanding Provider Edge Link Protection in Layer 3 VPNs on page 60](#)
- [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 61](#)

## Understanding Provider Edge Link Protection in Layer 3 VPNs

In an MPLS service provider network, a customer can have dual-homed CE routers that are connected to the service provider through different PE routers. This setup enables load balancing of traffic in the service provider network. However, this can lead to disruption in traffic if the link between a CE router and a PE router goes down. Hence, a precomputed protection path should be configured such that if a link between a CE router and a PE router goes down, the protection path (also known as the backup path) between the CE router and an alternate PE router can be used.

To configure a path to be a protection path, use the **protection** statement at the **[edit routing-instances *instance-name* protocols bgp family inet unicast]** hierarchy level:

```
routing-instances {
  customer {
    instance-type vrf;
    ...
    protocols {
      bgp {
        type external;
        ...
        family inet {
          unicast {
            protection;
          }
        }
        family inet6 {
          unicast {
            protection;
          }
        }
      }
    }
  }
}
```

The **protection** statement indicates that protection is required on prefixes received from the particular neighbor or family. After protection is enabled for a given family, group, or neighbor, protection entries are added for prefixes or next hops received from the given peer.



**NOTE:** A protection path can be selected only if the best path has already been installed by BGP in the forwarding table. This is because a protection path cannot be used as the best path.



**NOTE:** The option **vrf-table-label** must be configured under the **[routing-instances *instance-name*]** hierarchy for the routers that have protected PE-CE links. This applies to Junos OS Releases 12.3 through 13.2 inclusive.

The protection path selection takes place based on the value of two state flags:

- The **ProtectionPath** flag indicates paths requesting protection.
- The **ProtectionCand** flag indicates the route entry that can be used as a protection path.



NOTE:

- Provider edge link protection is configured only for external peers.
- If provider edge link protection is configured with the **equal-external-internal** multipath statement, multipath takes precedence over protection.

## Example: Configuring Provider Edge Link Protection in Layer 3 VPNs

This example shows how to configure a provider edge protection path that can be used in case of a link failure in an MPLS network.

- [Requirements on page 61](#)
- [Overview on page 61](#)
- [Configuration on page 62](#)
- [Verification on page 71](#)

### Requirements

This example uses the following hardware components, software components and configuration options:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 12.3 through 13.2 inclusive
- The option **vrf-table-label** must be enabled at the **[routing-instances *instance-name*]** hierarchy level for routers with protected PE-CE links.

### Overview

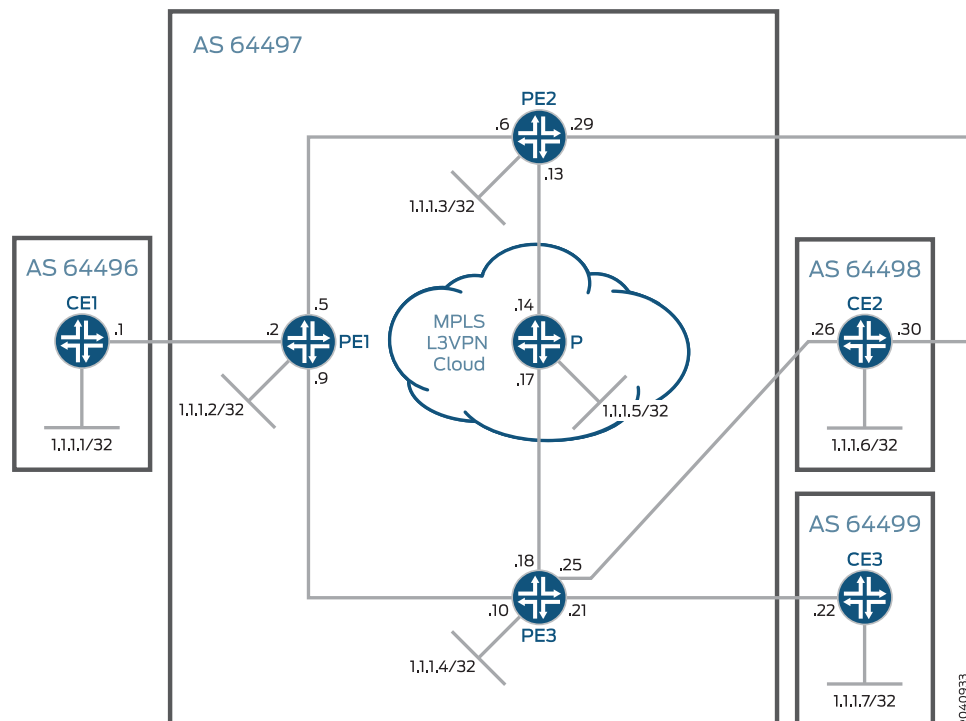
The following example shows how to configure provider edge link protection in a Layer 3 VPN.

### Topology

In this example, a Layer 3 VPN is set up by configuring three customer edge devices and three service provider edge devices in four autonomous systems. The CE devices are configured in AS 64496, AS 64498, and AS 64499. The PE devices are configured in AS 64497.

[Figure 14 on page 62](#) shows the topology used in this example.

Figure 14: Provider Edge Link Protection in a Layer 3 VPN



The aim of this example is to protect the provider edge link between Routers PE2 and CE2. Protection is configured on the backup link between Routers PE3 and CE2, such that the traffic can be routed through this link when the PE2-CE2 link goes down.

### Configuration

#### CLI Quick Configuration

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

```

Router CE1
set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.1/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::1/128
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 64496
set protocols bgp group toPE1 type external
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 peer-as 64497
set protocols bgp group toPE1 neighbor 10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

Router PE1
set interfaces ge-2/0/0 unit 0 description toCE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.2/30

```

```

set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE2
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.5/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toPE3
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.9/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.2/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::2/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.2
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/0.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE1 type external
set routing-instances radium protocols bgp group toCE1 peer-as 64496
set routing-instances radium protocols bgp group toCE1 neighbor 10.1.1.1
set policy-options policy-statement lb then load-balance per-packet

```

#### Router PE2

```

set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.6/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.13/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.29/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.3/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::3/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive

```

```

set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.3
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.30
set policy-options policy-statement lb then load-balance per-packet

```

#### Router PE3

```

set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.10/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.18/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.25/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces ge-2/0/3 unit 0 description toCE3
set interfaces ge-2/0/3 unit 0 family inet address 10.1.1.21/30
set interfaces ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
set interfaces ge-2/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.4/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::4/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.4
set protocols bgp group toInternal neighbor 1.1.1.2

```

```

set protocols bgp group toInternal neighbor 1.1.1.3
set routing-options router-id 1.1.1.4
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium vrf-table-label
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium interface ge-2/0/3.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.26
set routing-instances radium protocols bgp group toCE2 family inet unicast protection
set routing-instances radium protocols bgp group toCE2 family inet6 unicast protection
set routing-instances radium protocols bgp group toCE3 type external
set routing-instances radium protocols bgp group toCE3 peer-as 64499
set routing-instances radium protocols bgp group toCE3 neighbor 10.1.1.22
set policy-options policy-statement lb then load-balance per-packet

```

**Router P**

```

set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.14/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.17/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.5/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::5/128
set routing-options router-id 1.1.1.5
set routing-options autonomous-system 64497
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5

```

**Router CE2**

```

set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.30/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.26/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.6/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::6/128
set routing-options router-id 1.1.1.6
set routing-options autonomous-system 64498
set protocols bgp group toAS2 type external
set protocols bgp group toAS2 export send-direct

```

```

set protocols bgp group toAS2 peer-as 64497
set protocols bgp group toAS2 neighbor 10.1.1.25
set protocols bgp group toAS2 neighbor 10.1.1.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

**Router CE3**

```

set interfaces ge-2/0/0 unit 0 description toPE3
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.22/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.7/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::7/128
set routing-options router-id 1.1.1.7
set routing-options autonomous-system 64499
set protocols bgp group toPE3 type external
set protocols bgp group toPE3 export send-direct
set protocols bgp group toPE3 peer-as 64497
set protocols bgp group toPE3 neighbor 10.1.1.21
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

### Configuring Provider Edge Link Protection in Layer 3 VPNs

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

To configure provider edge link protection:

1. Configure the router interfaces.

```

[edit interfaces]
user@PE3# set interfaces ge-2/0/0 unit 0 description toPE1
user@PE3# set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.10/30
user@PE3# set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64
eui-64
user@PE3# set interfaces ge-2/0/0 unit 0 family mpls

user@PE3# set interfaces ge-2/0/1 unit 0 description toP
user@PE3# set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.18/30
user@PE3# set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64
eui-64
user@PE3# set interfaces ge-2/0/1 unit 0 family mpls

user@PE3# set interfaces ge-2/0/2 unit 0 description toCE2
user@PE3# set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.25/30
user@PE3# set interfaces ge-2/0/2 unit 0 family inet6 address 12001:db8:0:25::/64
eui-64
user@PE3# set interfaces ge-2/0/2 unit 0 family mpls

user@PE3# set interfaces ge-2/0/3 unit 0 description toCE3
user@PE3# set interfaces ge-2/0/3 unit 0 family inet address 10.1.1.21/30
user@PE3# set interfaces ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64
eui-64
user@PE3# set interfaces ge-2/0/3 unit 0 family mpls

user@PE3# set interfaces lo0 unit 0 family inet address 1.1.1.4/32
user@PE3# set interfaces lo0 unit 0 family inet6 address 2001:db8::4/128

```

Similarly, configure the interfaces on all other routers.

2. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@PE3# set router-id 1.1.1.4
user@PE3# set autonomous-system 64497
```

Similarly, configure the router ID and AS number for all other routers. In this example, the router ID is chosen to be identical to the loopback address configured on the router.

3. Configure MPLS and LDP on all interfaces of Router PE3.

```
[edit protocols]
user@PE3# set mpls interface all
user@PE3# set ldp interface all
```

Similarly, configure other PE routers.

4. Configure an IGP on the core-facing interfaces of Router PE3.

```
[edit protocols ospf area 0.0.0.0]
user@PE3# set interface lo0.0 passive
user@PE3# set interface ge-2/0/1.0 metric 5
user@PE3# set interface ge-2/0/0.0 metric 10

[edit protocols ospf3 area 0.0.0.0]
user@PE3# set interface lo0.0 passive
user@PE3# set interface ge-2/0/1.0 metric 5
user@PE3# set interface ge-2/0/0.0 metric 10
```

Similarly, configure other PE routers.

5. Configure a policy that exports the routes from the routing table into the forwarding table on Router PE3.

```
[edit policy-options]
user@PE3# set policy-statement lb then load-balance per-packet

[edit routing-options]
user@PE3# set forwarding-table export lb
```

Similarly, configure other PE routers.

6. Configure BGP on Router CE2, and include a policy for exporting routes to and from the service provider network.

```
[edit policy-options]
user@CE2# set policy-statement send-direct from protocol direct
user@CE2# set policy-statement send-direct then accept

[edit protocols bgp group toAS2]
user@CE2# set type external
user@CE2# set export send-direct
user@CE2# set peer-as 64497
user@CE2# set neighbor 10.1.1.25
user@CE2# set neighbor 10.1.1.29
```

Similarly, configure other CE routers.

7. Configure BGP on Router PE3 for routing within the provider core.

```
[edit protocols bgp group toInternal]
user@PE3# set type internal
user@PE3# set family inet-vpn unicast
user@PE3# set family inet6-vpn unicast
user@PE3# set multipath
user@PE3# set local-address 1.1.1.4
user@PE3# set neighbor 1.1.1.2
user@PE3# set neighbor 1.1.1.3
```

Similarly, configure other PE routers.

8. Configure the Layer 3 VPN routing instance on Router PE3.

```
[set routing-instances radium]
user@PE3# set instance-type vrf
user@PE3# set vrf-table-label
user@PE3# set interface ge-2/0/2.0
user@PE3# set interface ge-2/0/3.0
user@PE3# set route-distinguisher 64497:1
user@PE3# set vrf-target target:64497:1

[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set type external
user@PE3# set peer-as 64498
user@PE3# set neighbor 10.1.1.26

[edit routing-instances radium protocols bgp group toCE3]
user@PE3# set type external
user@PE3# set peer-as 64499
user@PE3# set neighbor 10.1.1.22
```

Similarly, configure other PE routers.

9. Configure provider edge link protection on the link between Routers PE3 and CE2.

```
[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set family inet unicast protection
user@PE3# set family inet6 unicast protection
```

## Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show policy-options**, **show protocols**, and **show routing-instances** commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show interfaces
ge-2/0/0 {
  unit 0 {
    description toPE1;
    family inet {
      address 10.1.1.10/30;
    }
    family inet6 {
      address 2001:db8:0:9::/64 {
        eui-64;
      }
    }
  }
}
```

```

        family mpls;
    }
}

ge-2/0/1 {
    unit 0 {
        description toP;
        family inet {
            address 10.1.1.18/30;
        }
        family inet6 {
            address 2001:db8:0:17::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

ge-2/0/2 {
    unit 0 {
        description toCE2;
        family inet {
            address 10.1.1.25/30;
        }
        family inet6 {
            address 2001:db8:0:25::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

ge-2/0/3 {
    unit 0 {
        description toCE3;
        family inet {
            address 10.1.1.21/30;
        }
        family inet6 {
            address 2001:db8:0:21::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

lo0 {
    unit 0 {
        family inet {
            address 1.1.1.4/32;
        }
        family inet6 {
            address 2001:db8::4/128;
        }
    }
}

```

```
user@PE3# show routing-options
```

```
router-id 1.1.1.4;
autonomous-system 64497;
forwarding-table {
    export lb;
}

user@PE3# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}

user@PE3# show protocols
mpls {
    interface all;
}
bgp {
    group toInternal {
        type internal;
        local-address 1.1.1.4;
        family inet-vpn {
            unicast;
        }
        family inet6-vpn {
            unicast;
        }
        multipath;
        neighbor 1.1.1.2;
        neighbor 1.1.1.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
        interface ge-2/0/0.0 {
            metric 10;
        }
    }
}
ospf3 {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
        interface ge-2/0/0.0 {
            metric 10;
        }
    }
}
ldp {
    interface all;
}
```

```

user@PE3# show routing-instances
radium {
    instance-type vrf;
    interface ge-2/0/2.0;
    interface ge-2/0/3.0;
    route-distinguisher 64497:1;
    vrf-target target:64497:1;
    protocols {
        bgp {
            group toCE2 {
                type external;
                family inet {
                    unicast {
                        protection;
                    }
                }
                family inet6 {
                    unicast {
                        protection;
                    }
                }
                peer-as 64498;
                neighbor 10.1.1.26;
            }
            group toCE3 {
                type external;
                peer-as 64499;
                neighbor 10.1.1.22;
            }
        }
    }
}

```

Run these commands on all other routers to confirm the configurations. If you are done configuring the routers, enter **commit** from configuration mode.

## Verification

Confirm that the configuration is working properly.

- [Verifying BGP on page 71](#)
- [Verifying Provider Edge Link Protection on page 73](#)

### Verifying BGP

**Purpose** Verify that BGP is functional in the Layer 3 VPN.

**Action** From operational mode on Router PE3, run the **show route protocol bgp** command.

```

user@PE3> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

```

```

1.1.1.1/32      *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                 AS path: 64496 I, validation-state: unverified
                 > to 10.1.1.9 via ge-2/0/0.0, Push 299792
1.1.1.6/32      @[BGP/170] 00:09:40, localpref 100
                 AS path: 64498 I, validation-state: unverified
                 > to 10.1.1.26 via ge-2/0/2.0
                 [BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                 AS path: 64498 I, validation-state: unverified
                 > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
1.1.1.7/32      *[BGP/170] 00:09:26, localpref 100
                 AS path: 64499 I, validation-state: unverified
                 > to 10.1.1.22 via ge-2/0/3.0
10.1.1.0/30     *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                 AS path: I, validation-state: unverified
                 > to 10.1.1.9 via ge-2/0/0.0, Push 299792
10.1.1.20/30    [BGP/170] 00:09:26, localpref 100
                 AS path: 64499 I, validation-state: unverified
                 > to 10.1.1.22 via ge-2/0/3.0
10.1.1.24/30    [BGP/170] 00:09:40, localpref 100
                 AS path: 64498 I, validation-state: unverified
                 > to 10.1.1.26 via ge-2/0/2.0
10.1.1.28/30    *[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                 AS path: I, validation-state: unverified
                 > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)

                 [BGP/170] 00:09:40, localpref 100
                 AS path: 64498 I, validation-state: unverified
                 > to 10.1.1.26 via ge-2/0/2.0

```

mpls.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)  
 + = Active Route, - = Last Active, \* = Both

```

64497:1:1.1.1.1/32
                 *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                 AS path: 64496 I, validation-state: unverified
                 > to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:1.1.1.6/32
                 *[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                 AS path: 64498 I, validation-state: unverified
                 > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
64497:1:10.1.1.0/30
                 *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                 AS path: I, validation-state: unverified
                 > to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:10.1.1.28/30
                 *[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                 AS path: I, validation-state: unverified
                 > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)

```

inet6.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

radium.inet6.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)

The output shows all the BGP routes in the routing table of Router PE3. This indicates that BGP is functioning as required.

Similarly, run this command on other routers to check if BGP is operational.

**Meaning** BGP is functional in the Layer 3 VPN.

### *Verifying Provider Edge Link Protection*

**Purpose** Verify that the provider edge link between Routers PE2 and CE2 is protected.

**Action** To verify that provider edge link protection is configured correctly:

1. Confirm that a route on Router CE2 is advertised to Router PE3, directly and through Router PE2.

If the route is advertised correctly, you will see multiple paths for the route.

From operational mode on Router PE3, run the **show route destination-prefix** command.

```
user@PE3> show route 1.1.1.6
radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

1.1.1.6/32          @[BGP/170] 02:55:36, localpref 100
                   AS path: 64498 I, validation-state: unverified
                   > to 10.1.1.26 via ge-2/0/2.0
                   [BGP/170] 00:10:13, localpref 100, from 1.1.1.3
                   AS path: 64498 I, validation-state: unverified
                   > to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)

                   #[Multipath/255] 00:10:13
                   > to 10.1.1.26 via ge-2/0/2.0
                   to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)
```

The output verifies the presence of multiple paths from Router PE3 to the destination route, **1.1.1.6**, on Router CE2. The first path is directly through the PE3-CE2 link (**10.1.1.26**). The second path is through the provider core and PE2 (**10.1.1.17**).

2. Verify that the protection path is correctly configured by confirming that the weight for the active path being protected is **0x1**, and the weight for the protection candidate path is **0x4000**.

From operational mode on Router PE3, run the **show route destination-prefix extensive** command.

```
user@PE3> show route 1.1.1.6 extensive
radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
1.1.1.6/32 (3 entries, 2 announced)
    State: <CalcForwarding>
TSI:
KRT in-kernel 1.1.1.6/32 -> {list:10.1.1.26, indirect(1048584)}
Page 0 idx 1 Type 1 val 9229c38
    Nexthop: Self
    AS path: [64497] 64498 I
    Communities:
Page 0 idx 2 Type 1 val 9229cc4
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [64497] 64498 I
    Communities: target:64497:1
Path 1.1.1.6 from 10.1.1.26 Vector len 4. Val: 1 2
```

```

@BGP      Preference: 170/-101
          Next hop type: Router, Next hop index: 994
          Address: 0x9240a74
          Next-hop reference count: 5
          Source: 10.1.1.26
Next hop: 10.1.1.26 via ge-2/0/2.0, selected
          Session Id: 0x200001
          State: <Active Ext ProtectionPath ProtectionCand>
          Peer AS: 64498
          Age: 2:55:54
          Validation State: unverified
          Task: BGP_64498.10.1.1.26+52214
          Announcement bits (1): 2-BGP_RT_Background
          AS path: 64498 I
          Accepted
          Localpref: 100
          Router ID: 1.1.1.6
BGP      Preference: 170/-101
          Route Distinguisher: 64497:1
          Next hop type: Indirect
          Address: 0x92413a8
          Next-hop reference count: 6
          Source: 1.1.1.3
          Next hop type: Router, Next hop index: 1322
Next hop: 10.1.1.17 via ge-2/0/1.0, selected
          Label operation: Push 299840, Push 299776(top)
          Label TTL action: prop-ttl, prop-ttl(top)
          Session Id: 0x200005
          Protocol next hop: 1.1.1.3
          Push 299840
          Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
          State: <Secondary NotBest Int Ext ProtectionCand>
          Inactive reason: Not Best in its group - Interior > Exterior > Exterior via

Interior
          Local AS: 64497 Peer AS: 64497
          Age: 10:31      Metric2: 1
          Validation State: unverified
          Task: BGP_64497.1.1.1.3+179
          Local AS: 64497 Peer AS: 64497
          Age: 10:31      Metric2: 1
          Validation State: unverified
          Task: BGP_64497.1.1.1.3+179
          AS path: 64498 I
          Communities: target:64497:1
          Import Accepted
          VPN Label: 299840
          Localpref: 100
          Router ID: 1.1.1.3
          Primary Routing Table bgp.l3vpn.0
          Indirect next hops: 1
            Protocol next hop: 1.1.1.3 Metric: 1
            Push 299840
            Indirect next hop: 94100ec 1048584 INH Session ID:

0x20000b
            Indirect path forwarding next hops: 1
              Next hop type: Router
              Next hop: 10.1.1.17 via ge-2/0/1.0
              Session Id: 0x200005
            1.1.1.3/32 Originating RIB: inet.3
              Metric: 1                      Node path count: 1
              Forwarding nexthops: 1

```

```

Nexthop: 10.1.1.17 via ge-2/0/1.0
#Multipath Preference: 255
Next hop type: List, Next hop index: 1048585
Address: 0x944c154
Next-hop reference count: 2
Next hop: ELNH Address 0x9240a74 weight 0x1, selected
equal-external-internal-type external
Next hop type: Router, Next hop index: 994
Address: 0x9240a74
Next-hop reference count: 5
Next hop: 10.1.1.26 via ge-2/0/2.0
Next hop: ELNH Address 0x92413a8 weight 0x4000
equal-external-internal-type internal
Next hop type: Indirect
Address: 0x92413a8
Next-hop reference count: 6
Protocol next hop: 1.1.1.3
Push 299840
Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b

Next hop type: Router, Next hop index: 1322
Address: 0x9241310
Next-hop reference count: 4
Next hop: 10.1.1.17 via ge-2/0/1.0
Label operation: Push 299840, Push 299776(top)
Label TTL action: prop-ttl, prop-ttl(top)
State: <ForwardingOnly Int Ext>
Inactive reason: Forwarding use only
Age: 10:31
Validation State: unverified
Task: RT
Announcement bits (1): 0-KRT
AS path: 64498 I

```

The output shows that the weight (**0x4000**) assigned to the PE3-CE2 path is greater than the weight (**0x1**) assigned to the PE2-CE2 path. This confirms that the PE2-CE2 path is protected by the PE3-CE2 path.

**Meaning** The provider edge link between Routers PE2 and CE2 is protected.

## Understanding Provider Edge Link Protection for BGP Labeled Unicast Paths

In MPLS service provider networks, when Layer 3 VPNs are used for carrier-of-carriers deployments, the protocol used to link the customer edge (CE) routers in one autonomous system (AS) and a provider edge (PE) router in another AS is BGP labeled-unicast. Reroute solutions between ASs are essential to help service providers ensure that network outages will have minimal impact on the data flows through the networks. A service provider that is a customer of another service provider can have different CE routers that are connected to the other service provider through different PE routers. This setup enables load balancing of traffic. However, this can lead to disruption in traffic if the link between one CE router and a PE router goes down. Therefore, a precomputed protection path should be configured such that if a link between a CE router and a PE router goes down, the protection path (also known as the backup path) between the other CE router and the alternative PE router can be used.

To configure a labeled-unicast path to be a protection path, use the **protection** statement at the **[edit routing-instances *instance-name* protocols bgp family inet labeled-unicast]** hierarchy level:

```
routing-instances {
  customer {
    instance-type vrf;
    ...
    protocols {
      bgp {
        family inet {
          labeled-unicast {
            protection;
          }
        }
        family inet6 {
          labeled-unicast {
            protection;
          }
        }
        type external;
        ...
      }
    }
  }
}
```

The **protection** statement indicates that protection is desired on prefixes received from the particular neighbor or family. After protection is enabled for a given family, group, or neighbor, protection entries are added for prefixes or next hops received from the given peer.



**NOTE:** A protection path can be selected only if the best path has already been installed by BGP in the forwarding table. This is because a protection path cannot be used as the best path.

To minimize packet loss when the protected path is down, also use the `per-prefix-label` statement at the `[edit routing-instances instance-name protocols bgp family inet labeled-unicast]` hierarchy level. Set this statement on every PE router within the AS containing the protected path.

The protection path selection takes place based on the value of two state flags:

- The **ProtectionPath** flag indicates paths desiring protection.
- The **ProtectionCand** flag indicates the route entry that can be used as a protection path.



**NOTE:**

- Provider edge link protection is configured only for external peers.
- If provider edge link protection is configured with the `equal-external-internal` multipath statement, multipath takes precedence over protection.

**Related  
Documentation**

- [Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths on page 332](#)

## Egress Protection for BGP Labeled Unicast

When network node or link failures occur, it takes some time to restore service using traditional routing table convergence. Local repair procedures can provide much faster restoration by establishing local protection as close to a failure as possible. Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or autonomous systems (ASs). If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

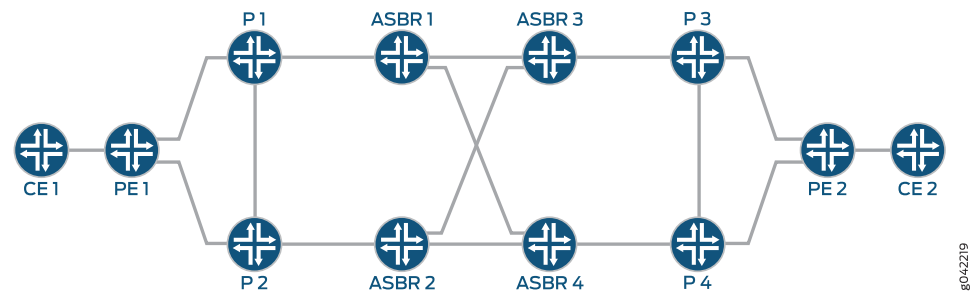
To provide egress protection for BGP labeled unicast, the protector node must create a backup state for downstream destinations before the failure happens. The basic idea of the solution is that the protector node constructs a forwarding state associated with the protected node and relays the MPLS labels assigned by the protected node further downstream to the final destination.

This feature supports the applications Inter-AS Option C and Seamless MPLS.

*Inter-AS Option C*—BGP labeled unicast provides end-to-end transport label-switched paths (LSPs) by stitching the intra-AS LSPs together. AS boundary routers run EBGp to other AS boundary routers to exchange labels for /32 PE loopback routes. IBGP runs between the provider edge router and AS boundary routers within each AS. In

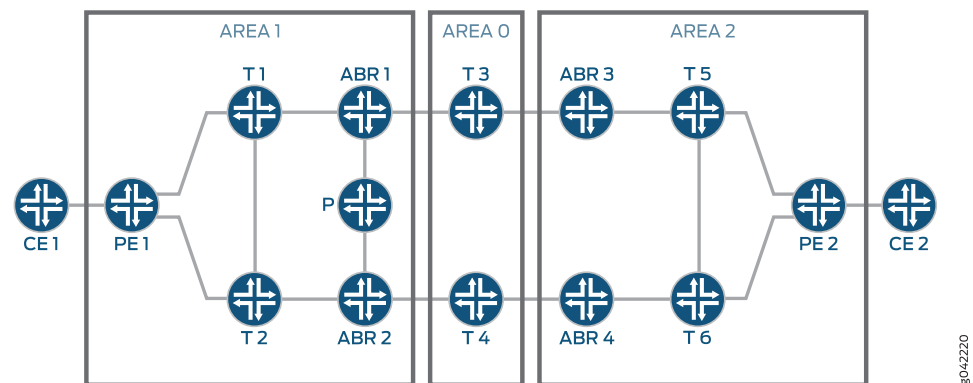
Figure 15 on page 78, the traffic goes from CE1 to CE2. ASBR1 is the protected AS boundary router, ASBR2 is the protector, and Device P1 is the point of local repair (PLR). The primary path is chosen from PE1 to PE2 over ASBR1 and ASBR3. When ASBR1 fails, Router P1 detects the ASBR1 failure and forwards the traffic to ASBR2, which provides backup service and forwards the traffic downstream.

Figure 15: Inter-AS Option C



*Seamless MPLS*—BGP labeled unicast provides end-to-end transport LSPs by stitching the intra-area/level LSPs. Area border routers (ABRs) run BGP labeled unicast to other ABRs to exchange labels for /32 PE loopback routes. In Figure 16 on page 78, the traffic goes from Device CE1 to Device CE2. ABR1 is the protected ABR, ABR2 is the protector, and T1 is the PLR. The primary path is chosen from PE1 to PE2 over ABR1 and ABR3. When ABR1 fails, Router T1 detects the ABR1 failure and forwards the traffic to ABR2, which provides backup service and forwards the traffic downstream.

Figure 16: Seamless MPLS



In each of these applications, the protected node advertises a primary BGP labeled unicast route that needs protection. When fast protection is enabled, BGP advertises the label routes with a special address as the next hop. This special address is a context identifier that is configured through the CLI. The protected node also advertises the context identifier in IGP and a NULL label in LDP for the context identifier.

The backup node advertises backup BGP labeled unicast routes for the protected routes. The protector node forwards traffic to the backup node using the labels advertised by the backup node.

The protector node provides the backup service by cross-connecting the labels originated by the protected node and the labels originated by the backup node. The protector node

forwards the traffic to the backup node in case of failure of the protected node. The protector node advertises the same context-identifier into IGP with high metric. Also, it advertises a real label in LDP for the context identifier. The protector node listens for the BGP labeled unicast routes advertised by both the protected node and backup node and populates the context label table and backup FIB. When traffic with the real context LDP label arrives, the lookup is done in the context of a protected node. The protector node often acts as the backup node.

The PLR detects the protected node failure and forwards the MPLS traffic to the protector node. The high IGP metric along with the LDP label advertised by the protector node ensure that the PLR uses the protector node as an LDP backup LSP.

There are two supported protection types: collocated protector and centralized protector. In the collocated type, the protector node is also the backup node. In the centralized type, the backup node is different from the protector node.

**Related  
Documentation**

- [Configuring Egress Protection for BGP Labeled Unicast on page 79](#)
- [Example: Configuring Egress Protection for BGP Labeled Unicast on page 345](#)
- [egress-protection on page 524](#)

---

## Configuring Egress Protection for BGP Labeled Unicast

---

Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or ASs. If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

Before configuring egress protection for BGP labeled unicast, ensure that all routers in the AS or area are running Junos OS 14.1 or a later release.

To configure egress protection for BGP labeled unicast:

1. Add the following configuration to the *protected* router:

```
[edit protocols]
  mpls {
    egress-protection {
      context-identifier context-id {
        primary;
      }
    }
  }
  bgp {
    group group-name {
      type internal;
      family inet {
        labeled-unicast {
          egress-protection {
            context-identifier context-id;
          }
        }
      }
    }
  }
}
```

2. Add the following configuration to the *protector* router:

```
[edit protocols]
  mpls {
    egress-protection {
      context-identifier context-id {
        protector;
      }
    }
  }
  bgp {
    group group-name {
      type internal;
      family inet {
        labeled-unicast {
          egress-protection;
        }
      }
    }
  }
}
```

3. Add the following configuration to the *PLR* (point of local repair) router:

```
[edit protocols]
  mpls {
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  isis {
    backup-spf-options per-prefix-calculation;
    level 1 disable;
    interface all {
      node-link-protection;
    }
  }
}
```

```

ldp {
  track-igp-metric;
  interface all;
  interface fxp0.0 {
    disable;
  }
}

```

4. Run **show bgp neighbor** on the protected router to verify that egress protection is enabled, for example:

```

user@host# run show bgp neighbor
Peer: 192.0.2.2+179 AS 65536 Local: 192.0.2.1+59264 AS 65536
Type: Internal State: Established Flags: <ImportEval Sync>
Last State: OpenConfirm Last Event: RecvKeepAlive
Last Error: None
Options: <Preference LocalAddress KeepAll AddressFamily Rib-group Refresh>
Address families configured: inet-label-unicast
Local Address: 192.0.2.1 Holdtime: 90 Preference: 170
NLRI configured with egress-protection: inet-label-unicast
Egress-protection NLRI inet-label-unicast
Number of flaps: 0

```

- Related Documentation**
- [Egress Protection for BGP Labeled Unicast on page 77](#)
  - [Example: Configuring Egress Protection for BGP Labeled Unicast on page 345](#)
  - [egress-protection \(MPLS\) on page 522](#)

## Configuring a Label Allocation and Substitution Policy for VPNs

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

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

```

label {
  allocation label-allocation-policy;
}

```

You can configure this statement at the following hierarchy levels:

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

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

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

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

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

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

You can configure this statement at the following hierarchy levels:

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

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

---

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

---

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

Doing this is useful for troubleshooting:

- It allows you to ping a remote CE router from a local PE router in a Layer 3 VPN. For more information, see [“Example: Troubleshooting Layer 3 VPNs” on page 574](#).
- It ensures that a path maximum transmission unit (MTU) check on traffic originating on a VRF or virtual-router routing instance functions properly. For more information, see [Configuring Path MTU Checks for VPNs](#).

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

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

- If you configure Filter A on the default loopback interface and Filter B on the VRF loopback interface, the VRF routing instance uses Filter B.
- If you configure Filter A on the default loopback interface but do not configure a filter on the VRF loopback interface, the VRF routing instance does not use a filter.
- If you configure Filter A on the default loopback interface but do not even configure a VRF loopback interface, the VRF routing instance uses Filter A.

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

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

You can include this statement at the following hierarchy levels:

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

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

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

You can include this statement at the following hierarchy levels:

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

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

```
interface lo0.n;
```

You can include this statement at the following hierarchy levels:

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

For more information about how to configure firewall filters, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices*.

## Configuring Multicast Layer 3 VPNs

---

You can configure two types of multicast Layer 3 VPNs using the Junos OS:

- Draft Rosen multicast VPNs—Draft Rosen multicast VPNs are described in RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)* and based on Section Two of the IETF Internet draft draft-rosen-vpn-mcast-06.txt, *Multicast in MPLS/BGP VPNs* (expired April 2004).
- Next generation multicast VPNs—Next generation multicast VPNs are described in Internet drafts draft-ietf-l3vpn-2547bis-mcast-bgp-03.txt, *BGP Encodings for Multicast in MPLS/BGP IP VPNs* and draft-ietf-l3vpn-2547bis-mcast-02.txt, *Multicast in MPLS/BGP IP VPNs*.

This section describes how to configure draft Rosen multicast VPNs. This information is provided to you in case you already have dual PIM multicast VPNs configured on your network. For information about BGP MPLS multicast VPNs (also known as next generation multicast VPNs), see *MBGP Multicast VPN Sites*.



**NOTE:** Draft-rosen multicast VPNs are not supported in a logical system environment even though the configuration statements can be configured under the logical-systems hierarchy.

You can configure a Layer 3 VPN to support multicast traffic using the Protocol Independent Multicast (PIM) routing protocol. To support multicast, you need to configure PIM on routers within the VPN and within the service provider's network.

Each PE router configured to run multicast over Layer 3 VPNs must have a Tunnel Services PIC. A Tunnel Services PIC is also required on the P routers that act as rendezvous points (RPs). Tunnel Services PICs are also needed on all the CE routers acting as designated routers (first-hop/last-hop routers) or as RPs, just as they are in non-VPN PIM environments.

Configure the master PIM instance at the **[edit protocols pim]** hierarchy level on the CE and PE routers. This master PIM instance configuration on the PE router should match the configuration on the service providers core routers.

You also need to configure a PIM instance for the Layer 3 VPN at the **[edit routing-instances routing-instance-name protocols pim]** hierarchy level on the PE router. This creates a PIM instance for the indicated routing instance. The configuration of the PIM instance on the PE router should match the PIM instance configured on the CE router the PE router is connected to.

For information about how to configure PIM, see the *Multicast Protocols Feature Guide for Routing Devices*.

Include the **vpn-apply-export** statement to configure the group address designated for the VPN in the service provider's network. This address must be unique for each VPN and

configured on the VRF routing instance of all PE routers connecting to the same VPN. It ensures that multicast traffic is transmitted only to the specified VPN.

Include the **vpn-apply-export** statement:

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

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

You can include this statement at the following hierarchy levels:

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

The rest of the Layer 3 VPN configuration for multicast is conventional and is described in other sections of this manual. Most of the specific configuration tasks needed to activate multicast in a VPN environment involve PIM. For more information about how to configure PIM and multicast in Junos, including an example of how to configure multicast over Layer 3 VPNs, see the *Multicast Protocols Feature Guide for Routing Devices*.

## Configuring Packet Forwarding for Layer 3 VPNs

You can configure the router to support packet forwarding for IPv4 traffic in Layer 2 and Layer 3 VPNs. Packet forwarding is handled in one of the following ways, depending on the type of helper service configured:

- BOOTP service—Clients send Bootstrap Protocol (BOOTP) requests through the router configured with BOOTP service to a server in the specified routing instance. The server recognizes the client address and sends a response back to the router configured with BOOTP service. This router forwards the reply to the correct client address in the specified routing instance.
- Other services—Clients send requests through the router configured with the service to a server in the specified routing instance. The server recognizes the client address and sends a response to the correct client address in the specified routing instance.

To enable packet forwarding for VPNs, include the **helpers** statement:

```
helpers {
  service {
    description description-of-service;
    server {
      address address {
        routing-instance routing-instance-names;
      }
    }
  }
  interface interface-name {
    description description-of-interface;
    no-listen;
    server {
      address address {
```

```

        routing-instance routing-instance-names;
    }
}
}
}

```

You can include this statement at the following hierarchy levels:

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



**NOTE:** You can enable packet forwarding for multiple VPNs. However, the client and server must be within the same VPN. Any Juniper Networks routing platforms with packet forwarding enabled along the path between the client and server must also reside within the same VPN.

The address and routing instance together constitute a unique server. This has implications for routers configured with BOOTP service, which can accept multiple servers.

For example, a BOOTP service can be configured as follows:

```

[edit forwarding-options helpers bootp]
server address 10.2.3.4 routing-instance [instance-A instance-B];

```

Even though the addresses are identical, the routing instances are different. A packet coming in for BOOTP service on **instance-A** is forwarded to **10.2.3.4** in the **instance-A** routing instance, while a packet coming in on **instance-B** is forwarded in the **instance-B** routing instance. Other services can only accept a single server, so this configuration does not apply in those cases.

For more information about the statements configured at the [edit forwarding-options] hierarchy level, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices*.

## Configuring GRE Tunnels for Layer 3 VPNs

Junos OS allows you to configure a generic routing encapsulation (GRE) tunnel between the PE and CE routers for a Layer 3 VPN. The GRE tunnel can have one or more hops. You can configure the tunnel from the PE router to a local CE router (as shown in [Figure 17 on page 87](#)) or to a remote CE router (as shown in [Figure 18 on page 87](#)).

Figure 17: GRE Tunnel Configured Between the Local CE Router and the PE Router

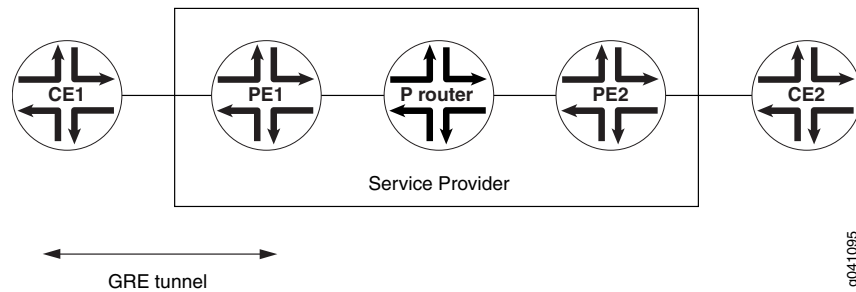
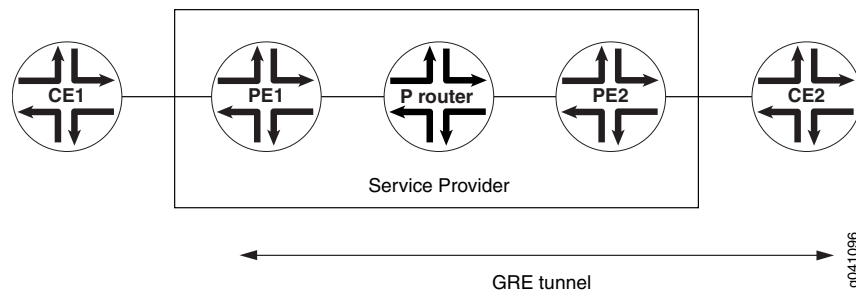


Figure 18: GRE Tunnel Configured Between the Remote CE Router and the PE Router



For more information about how to configure tunnel interfaces, see the *Junos OS Services Interfaces Library for Routing Devices*.

You can configure the GRE tunnels manually or configure the Junos OS to instantiate GRE tunnels dynamically.

The following sections describe how to configure GRE tunnels manually and dynamically:

- [Configuring GRE Tunnels Manually Between PE and CE Routers on page 87](#)
- [Configuring GRE Tunnels Dynamically on page 89](#)

## Configuring GRE Tunnels Manually Between PE and CE Routers

You can manually configure a GRE tunnel between a PE router and either a local CE router or a remote CE router for a Layer 3 VPN as explained in the following sections:

- [Configuring the GRE Tunnel Interface on the PE Router on page 87](#)
- [Configuring the GRE Tunnel Interface on the CE Router on page 88](#)

### Configuring the GRE Tunnel Interface on the PE Router

You configure the GRE tunnel as a logical interface on the PE router. To configure the GRE tunnel interface, include the **unit** statement:

```
unit logical-unit-number {
  tunnel {
    source source-address;
    destination destination-address;
    routing-instance {
```

```
        destination routing-instance-name;  
    }  
}  
family inet {  
    address address;  
}  
}
```

You can include this statement at the following hierarchy levels:

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

As part of the GRE tunnel interface configuration, you need to include the following statements:

- **source** *source-address*—Specify the source or origin of the GRE tunnel, typically the PE router.
- **destination** *destination-address*—Specify the destination or end point of the GRE tunnel. The destination can be a Provider router, the local CE router, or the remote CE router.

By default, the tunnel destination address is assumed to be in the default Internet routing table, *inet.0*. If the tunnel destination address is not in *inet.0*, you need to specify which routing table to search for the tunnel destination address by configuring the **routing-instance** statement. This is the case if the tunnel encapsulating interface is also configured under the routing instance.

- **destination** *routing-instance-name*—Specify the name of the routing instance when configuring the GRE tunnel interface on the PE router.

To complete the GRE tunnel interface configuration, include the **interface** statement for the GRE interface under the appropriate routing instance:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

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

---

### Configuring the GRE Tunnel Interface on the CE Router

You can configure either the local or the remote CE router to act as the endpoint for the GRE tunnel.

To configure the GRE tunnel interface on the CE router, include the **unit** statement:

```
unit logical-unit-number {  
    tunnel {  
        source address;  
        destination address;  
    }  
    family inet {
```

```

        address address;
    }
}

```

You can include this statement at the following hierarchy levels:

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

## Configuring GRE Tunnels Dynamically

When the router receives a VPN route to a BGP next hop address, but no MPLS path is available, a GRE tunnel can be dynamically generated to carry the VPN traffic across the BGP network. The GRE tunnel is generated and then its routing information is copied into the inet.3 routing table. IPv4 routes are the only type of routes supported for dynamic GRE tunnels. Also, the routing platform must have a tunnel PIC.



**NOTE:** When configuring a dynamic GRE tunnel to a remote CE router, do not configure OSPF over the tunnel interface. It creates a routing loop forcing the router to take the GRE tunnel down. The router attempts to reestablish the GRE tunnel, but will be forced to take it down again when OSPF becomes active on the tunnel interface and discovers a route to the tunnel endpoint. This is not an issue when configuring static GRE tunnels to a remote CE router.

To generate GRE tunnels dynamically, include the **dynamic-tunnels** statement:

```

dynamic-tunnels tunnel-name {
    destination-networks prefix;
    source-address address;
}

```

You can include this statement at the following hierarchy levels:

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

Specify the IPv4 prefix range (for example, 10/8 or 11.1/16) for the destination network by including the **destination-networks** statement. Only tunnels within the specified IPv4 prefix range are allowed to be initiated.

```

destination-networks prefix;

```

You can include this statement at the following hierarchy levels:

- [edit routing-options dynamic-tunnels *tunnel-name*]
- [edit logical-systems *logical-system-name* routing-options dynamic-tunnels *tunnel-name*]

Specify the source address for the GRE tunnels by including the **source-address** statement. The source address specifies the address used as the source for the local tunnel endpoint. This could be any local address on the router (typically the router ID or the loopback address).

```
source-address address;
```

You can include this statement at the following hierarchy levels:

- [edit routing-options dynamic-tunnels *tunnel-name*]
- [edit logical-systems *logical-system-name* routing-options dynamic-tunnels *tunnel-name*]

**Related  
Documentation**

- *Example: Configuring a Two-Tiered Virtualized Data Center for Large Enterprise Networks*

---

## Configuring an ES Tunnel Interface for Layer 3 VPNs

An ES tunnel interface allows you to configure an IP Security (IPsec) tunnel between the PE and CE routers of a Layer 3 VPN. The IPsec tunnel can include one or more hops.

The following sections explain how to configure an ES tunnel interface between the PE and CE routers of a Layer 3 VPN:

- [Configuring the ES Tunnel Interface on the PE Router on page 90](#)
- [Configuring the ES Tunnel Interface on the CE Router on page 91](#)

### Configuring the ES Tunnel Interface on the PE Router

To configure the ES tunnel interface on the PE router, include the **unit** statement:

```
unit logical-unit-number {  
  tunnel {  
    source source-address;  
    destination destination-address;  
  }  
  family inet {  
    address address;  
    ipsec-sa security-association-name;  
  }  
}
```

You can include this statement at the following hierarchy levels:

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

By default, the tunnel destination address is assumed to be in the default Internet routing table, inet.0. For IPsec tunnels using manual security association (SA), if the tunnel destination address is not in the default inet.0 routing table, you need to specify which routing table to search for the tunnel destination address by configuring the **routing-instance** statement. This is the case if the tunnel encapsulating interface is also configured under the routing instance.

```

unit logical-unit-number {
  tunnel {
    source address;
    destination address;
    routing-instance {
      destination routing-instance-name;
    }
    family inet {
      address address;
      ipsec-sa security-association-name;
    }
    family mpls;
  }
}

```

You can include these statements at the following hierarchy levels:

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



**NOTE:** For IPsec tunnels using dynamic SA, the tunnel destination address must be in the default Internet routing table, inet.0.

To complete the ES tunnel interface configuration, include the **interface** statement for the ES interface under the appropriate routing instance:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

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

## Configuring the ES Tunnel Interface on the CE Router

To configure the ES tunnel interface on the CE router, include the **unit** statement:

```

unit 0 {
  tunnel {
    source address;
    destination address;
  }
  family inet {
    address address;
    ipsec-sa security-association-name;
  }
}

```

You can include this statement at the following hierarchy levels:

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

For more information about how to configure tunnel interfaces, see the *Junos OS Services Interfaces Library for Routing Devices*.

For more information about how to configure IPsec interfaces, see the *Junos OS Administration Library for Routing Devices*.

## Configuring IPsec Tunnels Instead of MPLS LSPs Between PE Routers in Layer 3 VPNs

A conventional Layer 3 BGP/MPLS VPN requires the configuration of MPLS label-switched paths (LSPs) between the PE routers. When a PE router receives a packet from a CE router, it performs a lookup in a specific VRF table for the IP destination address and obtains a corresponding MPLS label stack. The label stack is used to forward the packet to the egress PE router, where the bottom label is removed and the packet is forwarded to the specified CE router.

You can provide Layer 3 BGP/MPLS VPN service without an MPLS backbone. Instead of configuring MPLS LSPs between the PE routers, you configure GRE and IPsec tunnels between the PE routers. The MPLS information for the VPN (the VPN label) is encapsulated within an IP header and an IPsec header. The source address of the IP header is the address of the ingress PE router. The destination address has the BGP next hop, the address of the egress PE router.



**NOTE:** The IPsec tunnel requires the use of an ES PIC. The GRE tunnel requires the use of a Tunnel Services PIC.

To configure IPsec between PE routers, follow these steps:

1. Configure an IPsec tunnel between the PE routers. The source address is that of the ingress PE router, and the destination address is that of the egress PE router:

```
es-interface-name {
  unit unit-number {
    tunnel {
      source source-address;
      destination destination-address;
    }
    family inet {
      ipsec-sa sa-esp-dynamic;
      address address;
    }
    family mpls;
  }
}
```

You can include these statements at the following hierarchy levels:

- **[edit interfaces]**
  - **[edit logical-systems *logical-system-name* interfaces]**
2. Configure IPsec on the PE router. For information about how to configure IPsec, see the *Junos OS Administration Library for Routing Devices*.

3. Configure a GRE tunnel between the PE routers. Again, the source address is that of the ingress PE router, and the destination address is that of the egress PE router:

```

gr-interface-name {
  unit unit-number {
    family inet {
      address address;
    }
    family mpls;
    tunnel {
      source source-address;
      destination destination-address;
    }
  }
}

```

You can include these statements at the following hierarchy levels:

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

4. Configure BGP between the PE routers:

```

bgp {
  group pe {
    type internal;
    local-address local-address;
    family inet {
      unicast;
    }
    family inet-vpn {
      unicast;
    }
    peer-as as-number;
    neighbor address;
  }
}

```

You can include these statements at the following hierarchy levels:

- **[edit protocols]**
- **[edit logical-systems *logical-system-name* protocols]**

5. Configure the routing instance:

```

instance-type vrf;
interface interface-name;
route-distinguisher address;
vrf-import import-policy-name;
vrf-export export-policy-name;
protocols {
  bgp {
    group routing-instance-name {
      type external;
      peer-as as-number;
      as-override;
      neighbor address;
    }
  }
}

```

```

    }
  }
}

```

You can include these statements at the following hierarchy levels:

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

6. Configure the policy options:

```

policy-statement import-policy-name {
  term 1 {
    from {
      protocol bgp;
      community community-name;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
policy-statement export-policy-name {
  term 1 {
    from protocol [ bgp direct ];
    then {
      community add community-name;
      accept;
    }
  }
  term 2 {
    then reject;
  }
}
community community-name members target:target;

```

You can include these statements at the following hierarchy levels:

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

7. Configure routing table groups to enable VPN route resolution in the inet.3 routing table:

```

interface-routes {
  rib-group inet if-rib;
}
rib inet.3 {
  static {
    route BGP-address-for-remote-PE next-hop gre-interface-name;
  }
}
rib-groups {
  if-rib {
    import-rib [ inet.0 inet.3 ];
  }
}

```

```
}
}
```

You can include these statements at the following hierarchy levels:

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

## Configuring Protocol-Independent Load Balancing in Layer 3 VPNs

Protocol-independent load balancing for Layer 3 VPNs allows the forwarding next hops of both the active route and alternative paths to be used for load balancing.

Protocol-independent load balancing works in conjunction with Layer 3 VPNs. It supports the load balancing of VPN routes independently of the assigned route distinguisher. When protocol-independent load balancing is enabled, both routes to other PE routers and routes to directly connected CE routers are load-balanced.

When load-balancing information is created for a given route, the active path is marked as **Routing Use Only** in the output of the **show route table** command.

The following sections describe how to configure protocol-independent load balancing and how this configuration can affect routing policies:

- [Configuring Load Balancing for Layer 3 VPNs on page 95](#)
- [Configuring Load Balancing and Routing Policies on page 96](#)

## Configuring Load Balancing for Layer 3 VPNs

The configuration of protocol-independent load balancing for Layer 3 VPNs is a little different for IPv4 versus IPv6:

- IPv4—You only need to configure the **multipath** statement at either the [edit routing-instances *routing-instance-name* routing-options] hierarchy level or the [edit routing-instances *routing-instance-name* routing-options rib *routing-table-name*] hierarchy level.
- IPv6—You need to configure the **multipath** statement at both the [edit routing-instances *routing-instance-name* routing-options] hierarchy level and the [edit routing-instances *routing-instance-name* routing-options rib *routing-table-name*] hierarchy level.



**NOTE:** You cannot configure the **multipath** statement and sub-statements at the same time that you have configured the **l3vpn** statement.

To configure protocol-independent load balancing for Layer 3 VPNs, include the **multipath** statement:

```
multipath {
  vpn-unequal-cost equal-external-internal;
}
```

When you include the **multipath** statement at the following hierarchy levels, protocol-independent load balancing is applied to the default routing table for that routing instance (*routing-instance-name.inet.0*):

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

When you include the **multipath** statement at the following hierarchy levels, protocol-independent load balancing is applied to the specified routing table:

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

The **vpn-unequal-cost** statement is optional:

- When you include it, protocol-independent load balancing is applied to VPN routes that are equal until the IGP metric with regard to route selection.
- When you do not include it, protocol-independent load balancing is applied to VPN routes that are equal until the router identifier with regard to route selection.

The **equal-external-internal** statement is also optional. When you include it, protocol-independent load balancing is applied to both internal and external BGP paths. You can configure this in conjunction with egress IP header filtering (enabled with the **vrf-table-label** statement). For more information, see [“Load Balancing and IP Header Filtering for Layer 3 VPNs” on page 59](#).



**NOTE:** You can include the **vpn-unequal-cost** **equal-external-internal** statement and the **l3vpn** statement at the [edit routing-options forwarding-options chained-composite-next-hop ingress] hierarchy level simultaneously. However, if you do this, EBGp does not work. This means that when there are both paths with chained next hops and paths with nonchained next hops as candidates for EBGp equal-cost multipath (ECMP), the paths using chained next hops are excluded. In a typical case, the excluded paths are the internal paths.

## Configuring Load Balancing and Routing Policies

If you enable protocol-independent load balancing for Layer 3 VPNs by including the **multipath** statement and if you also include the **load-balance per-packet** statement in the routing policy configuration, packets are not load-balanced.

For example, a PE router has the following VRF routing instance configured:

```
[edit routing-instances]
load-balance-example {
  instance-type vrf;
  interface fe-0/1/1.0;
```

```

interface fe-0/1/1.1;
route-distinguisher 2222:2;
vrf-target target:2222:2;
routing-options {
  multipath;
}
protocols {
  bgp {
    group group-example {
      import import-policy;
      family inet {
        unicast;
      }
      export export-policy;
      peer-as 4444;
      local-as 3333;
      multipath;
      as-override;
      neighbor 10.12.33.22;
    }
  }
}

```

The PE router also has the following policy statement configured:

```

[edit policy-options policy-statement export-policy]
from protocol bgp;
then {
  load-balance per-packet;
}

```

When you include the **multipath** statement in the VRF routing instance configuration, the paths are no longer marked as BGP paths but are instead marked as multipath paths. Packets from the PE router are not load-balanced.

To ensure that VPN load-balancing functions as expected, do not include the **from protocol** statement in the policy statement configuration. The policy statement should be configured as follows:

```

[edit policy-options policy-statement export-policy]
then {
  load-balance per-packet;
}

```

For more information about how to configure per-packet load balancing, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices*.

**Related Documentation**

- [VPN Per-Packet Load Balancing on page 21](#)

## Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes

By default, the third step of the algorithm that determines the active route evaluates the length of the AS path but not the contents of the AS path. In some VPN scenarios with

BGP multiple path routes, it can also be useful to compare the AS numbers of the AS paths and to have the algorithm select the route whose AS numbers match.

To configure the algorithm that selects the active path to evaluate the AS numbers in AS paths for VPN routes:

- Include the **as-path-compare** statement at the **[edit routing-instances routing-instance-name routing-options multipath]** hierarchy level.



**NOTE:** The **as-path-compare** statement is not supported for the default routing instance.

---

Related Documentation

- [as-path-compare on page 517](#)

---

## Configuring Traffic Policing in Layer 3 VPNs

---

You can use policing to control the amount of traffic flowing over the interfaces servicing a Layer 3 VPN. If policing is disabled on an interface, all the available bandwidth on a Layer 3 VPN tunnel can be used by a single CCC or TCC interface.

For more information about the **policer** statement, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices*.

To enable Layer 3 VPN policing on an interface, include the **policer** statement:

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

If you configure CCC encapsulation, you can include the **policer** statement at the following hierarchy levels:

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

If you configure TCC encapsulation, you can include the **policer** statement at the following hierarchy levels:

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

## Configuring BGP PIC Edge for MPLS Layer 3 VPNs

In an MPLS VPN Layer 3 environment, it is common for customers to multihome their networks to provide link redundancy. Although the interior gateway protocol (IGP) can provide fast convergence, in certain instances, the time to resolve a link failure and provide an alternate route can be time consuming. For example, a provider edge (PE) router might be configured with 200,000 or more IP prefixes, and a PE router failure could affect many of those prefixes.

BGP Prefix-Independent Convergence (PIC) Edge allows you to install a Layer 3 VPN route in the forwarding table as an alternate path, enabling fast failover when a PE router fails or you lose connectivity to a PE router. This already installed path is used until global convergence through the IGP is resolved. Using the alternative VPN route for forwarding until global convergence is complete reduces traffic loss.

BGP PIC Edge supports multiprotocol BGP IPv4 or IPv6 VPN network layer reachability information (NLRI) resolved using any of these IGP protocols:

- OSPF
- IS-IS
- LDP

BGP PIC Edge does not support multicast traffic.

Before you begin:

1. Configure LDP.
2. Configure an IGP, either OSPF or IS-IS.
3. Configure a Layer 3 VPN.
4. Configure multiprotocol BGP for either an IPv4 VPN or an IPv6 VPN.

To configure BGP PIC Edge in an MPLS Layer 3 VPN:

1. Enable BGP PIC Edge:

```
[edit routing-instances routing-instance-name routing-options]
user@host# set protect core
```



**NOTE:** The BGP PIC edge feature is supported only on MX Series 3D Universal Edge routers with MPC interfaces.

2. Configure per-packet load balancing:

```
[edit policy-options]
user@host# set policy-statement policy-name then load-balance per-packet
```

3. Apply the per-packet load balancing policy to routes exported from the routing table to the forwarding table:

```

asdfasdf
[edit routing-options forwarding-table]
user@host# set export policy-statement-name

```

4. Verify that BGP PIC Edge is working.

From operational mode, enter the **show route extensive** command:

```

user@host> show route 1.1.1.6 extensive
ed.inet.0: 6 destinations, 9 routes (6 active, 0 holddown, 0 hidden)
  1.1.1.6/32 (3 entries, 2 announced)
    State: <CalcForwarding>
    TSI:
    KRT in-kernel 1.1.1.6/32 -> {indirect(1048574), indirect(1048577)}
    Page 0 idx 0 Type 1 val 9219e30
    Nexthop: Self
    AS path: [2] 3 I
    Communities: target:2:1
    Path 1.1.1.6 from 1.1.1.4 Vector len 4. Val: 0
..
    #Multipath Preference: 255
    Next hop type: Indirect
    Address: 0x93f4010
    Next-hop reference count: 2
..
    Protocol next hop: 1.1.1.4
    Push 299824
    Indirect next hop: 944c000 1048574 INH Session ID: 0x3
    Indirect next hop: weight 0x1
    Protocol next hop: 1.1.1.5
    Push 299824
    Indirect next hop: 944c1d8 1048577 INH Session ID: 0x4
    Indirect next hop: weight 0x4000
    State: <ForwardingOnly Int Ext>
    Inactive reason: Forwarding use only
    Age: 25 Metric2: 15
    Validation State: unverified
    Task: RT
    Announcement bits (1): 0-KRT
    AS path: 3 I
    Communities: target:2:1

```

The output lines that contain **Indirect next hop: weight** follow next hops that the software can use to repair paths where a link failure occurs. The next-hop weight has one of the following values:

- 0x1 indicates active next hops.
- 0x4000 indicates passive next hops.



#### BEST PRACTICE:

On MX Series 3D Universal Edge Routers with Modular Port Concentrators (MPCs), we strongly recommend that you enable enhanced IP network services.

To enable enhanced IP network services:

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

#### Related Documentation

- [Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs on page 101](#)
- [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 59](#)

## Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs

This example shows how to configure BGP prefix-independent convergence (PIC) edge, which allows you to install a Layer 3 VPN route in the forwarding table as an alternate path. This enables fast failover when a provider edge (PE) router fails or you lose connectivity to a PE router. This already installed path is used until global convergence through the interior gateway protocol (IGP) is resolved. Using the alternative VPN route for forwarding until global convergence is complete reduces traffic loss.

- [Requirements on page 101](#)
- [Overview on page 101](#)
- [Configuration on page 102](#)
- [Verification on page 108](#)

### Requirements

No special configuration beyond device initialization is required before configuring this example.

This example uses the following hardware and software components:

- One MX Series 3D Universal Edge Routers with MPC interfaces to configure the BGP PIC edge feature.
- Five routers that can be a combination of M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers.
- Junos OS Release 13.2 or later on the device with BGP PIC edge configured.

### Overview

In an MPLS VPN Layer 3 environment, it is common for customers to multihomed their networks to provide link redundancy. Although the interior gateway protocol (IGP) can provide fast convergence, in certain instances, the time to resolve a link failure and provide an alternate route can be time consuming. For example, a provider edge (PE) router might be configured with 200,000 or more IP prefixes, and a PE router failure could affect many of those prefixes.

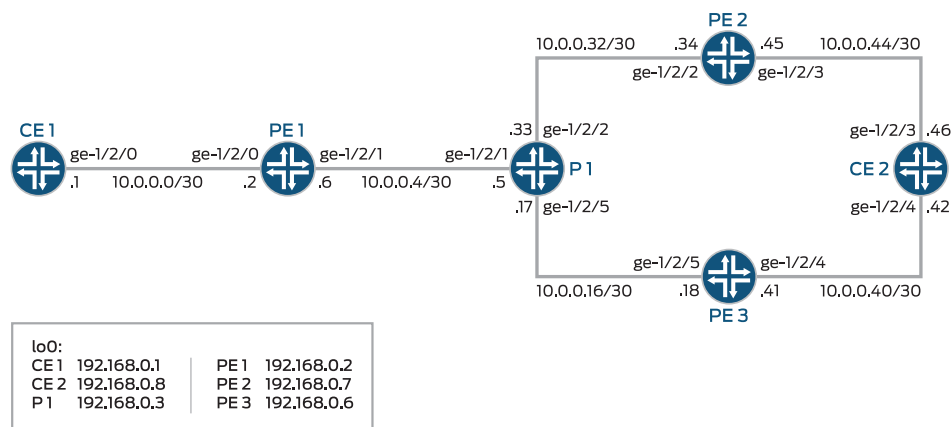
This example shows two customer edge (CE) routers, Device CE1 and Device CE2. Devices PE1, PE2, and PE3 are PE routers. Device P1 is a provider core router. Only Device PE1 has BGP PIC edge configured. The example uses the P1-PE2 link (P-PE) link to simulate the loss of a section of the network.

For testing, the address 172.16.1.5/24 is added as a loopback interface address on Device CE2. The address is announced to Device PE2 and Device PE3 and is relayed by way of internal BGP (IBGP) IBGP to Device PE1. On Device PE1, there are two paths to the 172.16.1.5/24 network. These are the primary and a backup path.

## Topology

Figure 19 on page 102 shows the sample network.

Figure 19: BGP PIC Edge Scenario



g041992

"CLI Quick Configuration" on page 102 shows the configuration for all of the devices in Figure 19 on page 102.

The section "Step-by-Step Procedure" on page 105 describes the steps on Device PE1.

## Configuration

### CLI Quick Configuration

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

- Device CE1**
- ```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ebgp type external
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 10.0.0.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options autonomous-system 101
```
- Device CE2**
- ```
set interfaces ge-1/2/4 unit 0 family inet address 10.0.0.42/30
set interfaces ge-1/2/3 unit 0 family inet address 10.0.0.46/30
set interfaces lo0 unit 0 family inet address 192.168.0.8/32
```

```

set interfaces lo0 unit 0 family inet address 172.16.1.5/24
set protocols bgp group ebgp type external
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 10.0.0.45
set protocols bgp group ebgp neighbor 10.0.0.41
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options autonomous-system 102

```

Device P1

```

set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/5 unit 0 family inet address 10.0.0.17/30
set interfaces ge-1/2/5 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.33/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols mpls interface ge-1/2/1.0
set protocols mpls interface ge-1/2/5.0
set protocols mpls interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ospf area 0.0.0.0 interface ge-1/2/5.0
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface ge-1/2/5.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-options autonomous-system 100

```

Device PE1

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.6/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols mpls interface ge-1/2/1.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.2
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.7
set protocols bgp group ibgp neighbor 192.168.0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement lb then load-balance per-packet
set policy-options policy-statement nhs then next-hop self
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/0.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 routing-options protect core
set routing-instances customer1 protocols bgp group ebgp type external
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.1
set routing-options router-id 192.168.0.2

```

```
set routing-options autonomous-system 100
set routing-options forwarding-table export lb
```

```
Device PE2
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.34/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/3 unit 0 family inet address 10.0.0.45/30
set interfaces lo0 unit 0 family inet address 192.168.0.7/32
set protocols mpls interface ge-1/2/2.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.7
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.2
set protocols bgp group ibgp neighbor 192.168.0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/3.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 protocols bgp group ebgp type external
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.46
set routing-options autonomous-system 100
```

```
Device PE3
set interfaces ge-1/2/5 unit 0 family inet address 10.0.0.18/30
set interfaces ge-1/2/5 unit 0 family mpls
set interfaces ge-1/2/4 unit 0 family inet address 10.0.0.41/30
set interfaces ge-1/2/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols mpls interface ge-1/2/5.0
set protocols mpls interface ge-1/2/4.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.6
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.7
set protocols bgp group ibgp neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface ge-1/2/5.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/5.0
set protocols ldp interface lo0.0
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/4.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 protocols bgp group ebgp type external
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.42
set routing-options autonomous-system 100
```

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE1# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
```

```
user@PE1# set ge-1/2/1 unit 0 family inet address 10.0.0.6/30
user@PE1# set ge-1/2/1 unit 0 family mpls
```

```
user@PE1# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure MPLS and LDP on the core-facing interfaces.

```
[edit protocols]
user@PE1# set mpls interface ge-1/2/1.0
```

```
user@PE1# set ldp interface ge-1/2/1.0
user@PE1# set ldp interface lo0.0
```

3. Configure an IGP on the core-facing interfaces.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface ge-1/2/1.0
user@PE1# set interface lo0.0 passive
```

4. Configure IBGP connections with the other PE devices.

```
[edit protocols bgp group ibgp]
user@PE1# set type internal
user@PE1# set local-address 192.168.0.2
user@PE1# set family inet unicast
user@PE1# set family inet-vpn unicast
user@PE1# set export nhs
user@PE1# set neighbor 192.168.0.7
user@PE1# set neighbor 192.168.0.6
```

5. Configure the load-balancing policy.

```
[edit policy-options policy-statement lb]
user@PE1# set then load-balance per-packet
```

6. (Optional) Configure a next-hop self policy.

```
[edit policy-options policy-statement nhs]
user@PE1# set then next-hop self
```

7. Configure the routing-instance to create the CE-PE EBGP connection.

```
[edit routing-instances customer1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 100:1
user@PE1# set vrf-target target:100:1
user@PE1# set protocols bgp group ebgp type external
```

```
user@PE1# set protocols bgp group ebgp neighbor 10.0.0.1
```

8. Enable the BGP PIC edge feature.

```
[edit routing-instances customer1]
user@PE1# set routing-options protect core
```

9. Apply the load-balancing policy.

```
[edit routing-options forwarding-table]
user@PE1# set export lb
```

10. Assign the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@PE1# set router-id 192.168.0.2
user@PE1# set autonomous-system 100
```

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

```
user@PE1# show interfaces
ge-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
}
ge-1/2/1 {
  unit 0 {
    family inet {
      address 10.0.0.6/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.2/32;
    }
  }
}

user@PE1# show protocols
mpls {
  interface ge-1/2/1.0;
}
bgp {
  group ibgp {
    type internal;
    local-address 192.168.0.2;
    family inet {
      unicast;
    }
  }
}
```

```
family inet-vpn {
    unicast;
}
export nhs;
neighbor 192.168.0.7;
neighbor 192.168.0.6;
}
}
ospf {
    area 0.0.0.0 {
        interface ge-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
}
ldp {
    interface ge-1/2/1.0;
    interface lo0.0;
}

user@PE1# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
policy-statement nhs {
    then {
        next-hop self;
    }
}

user@PE1# show routing-instances
customer1 {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 100:1;
    vrf-target target:100:1;
    routing-options {
        protect core;
    }
    protocols {
        bgp {
            group ebgp {
                type external;
                peer-as 101;
                neighbor 10.0.0.1;
            }
        }
    }
}

user@PE1# show routing-options
router-id 192.168.0.2;
autonomous-system 100;
forwarding-table {
```

```
export lb;
}
```

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

## Verification

Confirm that the configuration is working properly.

- [Displaying Extensive Route Information on page 108](#)
- [Displaying the Forwarding Table on page 110](#)
- [Displaying the OSPF Routes on page 111](#)

### Displaying Extensive Route Information

**Purpose** Confirm that BGP PIC Edge is working.

**Action** From Device PE1, run the **show route extensive table customer1.inet.0 172.16.1/24** command.

```
user@PE1> show route extensive table customer1.inet.0 172.16.1/24
```

```
customer1.inet.0: 7 destinations, 12 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (3 entries, 2 announced)
```

```
State: <CalcForwarding>
```

```
TSI:
```

```
KRT in-kernel 172.16.1.0/24 -> {indirect(262146), indirect(262142)}
```

```
Page 0 idx 0, (group ebgp type External) Type 1 val 0x950a62c (adv_entry)
```

```
Advertised metrics:
```

```
Nexthop: Self
```

```
AS path: [100] 102 I
```

```
Communities: target:100:1
```

```
Path 172.16.1.0 from 192.168.0.6 Vector len 4. Val: 0
```

```
@BCP Preference: 170/-101
```

```
Route Distinguisher: 100:1
```

```
Next hop type: Indirect
```

```
Address: 0x9514a74
```

```
Next-hop reference count: 7
```

```
Source: 192.168.0.6
```

```
Next hop type: Router, Next hop index: 990
```

```
Next hop: 10.0.0.5 via ge-1/2/1.0, selected
```

```
Label operation: Push 299824, Push 299856(top)
```

```
Label TTL action: prop-ttl, prop-ttl(top)
```

```
Load balance label: Label 299824: None; Label 299856: None;
```

```
Session Id: 0x280002
```

```
Protocol next hop: 192.168.0.6
```

```
Label operation: Push 299824
```

```
Label TTL action: prop-ttl
```

```
Load balance label: Label 299824: None;
```

```
Indirect next hop: 0x96bc104 262146 INH Session ID: 0x280006
```

```
State: <Secondary Active Int Ext ProtectionPath ProtectionCand>
```

```
Local AS: 100 Peer AS: 100
```

```
Age: 1:38:13 Metric2: 1
```

```
Validation State: unverified
```

```
Task: BGP_100.192.168.0.6+45824
```

```
Announcement bits (1): 1-BGP_RT_Background
```

```
AS path: 102 I
```

```
Communities: target:100:1
```

```
Import Accepted
```

```

VPN Label: 299824
Localpref: 100
Router ID: 192.168.0.6
Primary Routing Table bgp.13vpn.0
Indirect next hops: 1
    Protocol next hop: 192.168.0.6 Metric: 1
    Label operation: Push 299824
    Label TTL action: prop-ttl
    Load balance label: Label 299824: None;
    Indirect next hop: 0x96bc104 262146 INH Session ID:

0x280006
    Indirect path forwarding next hops: 1
        Next hop type: Router
        Next hop: 10.0.0.5 via ge-1/2/1.0
        Session Id: 0x280002
    192.168.0.6/32 Originating RIB: inet.3
        Metric: 1
        Node path count: 1
        Forwarding nexthops: 1
            Nexthop: 10.0.0.5 via ge-1/2/1.0
BCP Preference: 170/-101
Route Distinguisher: 100:1
Next hop type: Indirect
Address: 0x9515570
Next-hop reference count: 7
Source: 192.168.0.7
Next hop type: Router, Next hop index: 933
Next hop: 10.0.0.5 via ge-1/2/1.0, selected
Label operation: Push 299856, Push 299872(top)
Label TTL action: prop-ttl, prop-ttl(top)
Load balance label: Label 299856: None; Label 299872: None;
Session Id: 0x280002
Protocol next hop: 192.168.0.7
Label operation: Push 299856
Label TTL action: prop-ttl
Load balance label: Label 299856: None;
Indirect next hop: 0x96bc000 262142 INH Session ID: 0x280005
State: <Secondary NotBest Int Ext ProtectionPath ProtectionCand>

Inactive reason: Not Best in its group - Router ID
Local AS: 100 Peer AS: 100
Age: 1:38:13 Metric2: 1
Validation State: unverified
Task: BGP_100.192.168.0.7+10985
AS path: 102 I
Communities: target:100:1
Import Accepted
VPN Label: 299856
Localpref: 100
Router ID: 192.168.0.7
Primary Routing Table bgp.13vpn.0
Indirect next hops: 1
    Protocol next hop: 192.168.0.7 Metric: 1
    Label operation: Push 299856
    Label TTL action: prop-ttl
    Load balance label: Label 299856: None;
    Indirect next hop: 0x96bc000 262142 INH Session ID:

0x280005
    Indirect path forwarding next hops: 1
        Next hop type: Router
        Next hop: 10.0.0.5 via ge-1/2/1.0
        Session Id: 0x280002

```

```

192.168.0.7/32 Originating RIB: inet.3
Metric: 1                               Node path count: 1
Forwarding nexthops: 1
    Nexthop: 10.0.0.5 via ge-1/2/1.0
#Multipath Preference: 255
    Next hop type: Indirect
    Address: 0x9578010
    Next-hop reference count: 4
    Next hop type: Router, Next hop index: 990
    Next hop: 10.0.0.5 via ge-1/2/1.0, selected
    Label operation: Push 299824, Push 299856(top)
    Label TTL action: prop-ttl, prop-ttl(top)
    Load balance label: Label 299824: None; Label 299856: None;
    Session Id: 0x280002
    Next hop type: Router, Next hop index: 933
    Next hop: 10.0.0.5 via ge-1/2/1.0
    Label operation: Push 299856, Push 299872(top)
    Label TTL action: prop-ttl, prop-ttl(top)
    Load balance label: Label 299856: None; Label 299872: None;
    Session Id: 0x280002
    Protocol next hop: 192.168.0.6
    Label operation: Push 299824
    Label TTL action: prop-ttl
    Load balance label: Label 299824: None;
    Indirect next hop: 0x96bc104 262146 INH Session ID: 0x280006

Weight 0x1
    Protocol next hop: 192.168.0.7
    Label operation: Push 299856
    Label TTL action: prop-ttl
    Load balance label: Label 299856: None;
    Indirect next hop: 0x96bc000 262142 INH Session ID: 0x280005

Weight 0x4000
    State: <ForwardingOnly Int Ext>
    Inactive reason: Forwarding use only
    Age: 1:38:13    Metric2: 1
    Validation State: unverified
    Task: RT
    Announcement bits (1): 0-KRT
    AS path: 102 I
    Communities: target:100:1

```

**Meaning** The Indirect next hop output lines that contain weight follow next hops that the software can use to repair paths where a link failure occurs.

The next-hop weight has one of the following values:

- 0x1 indicates active next hops.
- 0x4000 indicates passive next hops.

### Displaying the Forwarding Table

**Purpose** Check the forwarding and kernel routing-table state by using **show route forwarding-table**.

**Action** From Device PE1, run the **show route forwarding-table table customer1 destination 172.16.1.0/24** command.

```
user@PE1> show route forwarding-table table customer1 destination 172.16.1.0/24
```

```
Routing table: customer1.inet
Internet:
Destination          Type RtRef Next hop          Type Index  NhRef Netif
172.16.1.0/24        user    0              10.0.0.5        ulst  262147    2
                    user    0              10.0.0.5        indr  262146    3
                    990    2 ge-1/2/1.0      Push 299824, Push 299856(top)
                    1000   2 ge-1/2/1.0      10.0.0.5        indr  262144    3
                    Push 300080, Push 299920(top)
```

**Meaning** In addition to the forwarding and kernel routing-table state, this command shows the unicast index (262147) used by the Packet Forwarding Engine.

### Displaying the OSPF Routes

**Purpose** Show the OSPF route state.

**Action** From Device PE1, run the **show (ospf | ospf3) route detail** command.

```
user@PE1> show ospf route detail
```

```
betsy@tp0:PE1> show ospf route detail
Topology default Route Table:
```

Prefix	Path	Route Type	NH Type	Metric	NextHop Interface	Nexthop Address/LSP
192.168.0.3	Intra	Router	IP	1	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.3, optional-capability 0x0						
192.168.0.6	Intra	Router	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.6, optional-capability 0x0						
192.168.0.7	Intra	Router	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.7, optional-capability 0x0						
10.0.0.4/30	Intra	Network	IP	1	ge-1/2/1.0	
area 0.0.0.0, origin 192.168.0.3, priority low						
10.0.0.16/30	Intra	Network	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.6, priority medium						
10.0.0.32/30	Intra	Network	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.7, priority medium						
192.168.0.2/32	Intra	Network	IP	0	lo0.0	
area 0.0.0.0, origin 192.168.0.2, priority low						
192.168.0.3/32	Intra	Network	IP	1	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.3, priority medium						
<b>192.168.0.6/32</b>	Intra	Network	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.6, priority medium						
<b>session-id: 2621446, version: 1</b>						
<b>192.168.0.7/32</b>	Intra	Network	IP	2	ge-1/2/1.0	10.0.0.5
area 0.0.0.0, origin 192.168.0.7, priority medium						
<b>session-id: 2621450, version: 1</b>						

**Meaning** The output shows the tracked session IDs for the loopback interface addresses on Devices PE2 and PE3.

- Related Documentation**
- [Configuring BGP PIC Edge for MPLS Layer 3 VPNs on page 99](#)
  - [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 59](#)

---

## Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs

For Layer 3 VPNs configured on Juniper Networks routers, Junos OS normally allocates one inner VPN label for each customer edge (CE)-facing virtual routing and forwarding (VRF) interface of a provider edge (PE) router. However, other vendors allocate one VPN label for each route learned over the CE-facing interfaces of a PE router. This practice increases the number of VPN labels exponentially, which leads to slow system processing and slow convergence time.

Next-hop chaining (also known as “chained composite next hop”) is a composition function that concatenates the partial rewrite strings associated with individual next hops to form a larger rewrite string that is added to a packet. By using this function, the number of routes with unique inner VPN labels that can be processed by a Juniper Networks router is increased substantially. Common route update elements associated with Layer 3 VPNs are combined, reducing the number of route updates and individual states the Juniper Networks router must maintain, and leading to enhanced scaling and convergence performance.

You can configure the router based on the number of VPN labels you want to manage and on whether or not you want to create chained composite next hops for IPv6 labeled routes:

- [Accepting Up to One Million Layer 3 VPN Route Updates on page 112](#)
- [Accepting More Than One Million Layer 3 VPN Route Updates on page 113](#)
- [Enabling Chained Composite Next Hops for IPv6 Labeled Unicast Routes on page 115](#)

### Accepting Up to One Million Layer 3 VPN Route Updates

For Juniper Networks routers participating in a mixed vendor network with up to one million Layer 3 VPN labels, include the **l3vpn** statement at the **[edit routing-options forwarding-table chained-composite-next-hop ingress]** hierarchy level. The **l3vpn** statement is disabled by default.



**BEST PRACTICE:** We recommend that you configure the **l3vpn** statement whenever you have deployed Juniper Networks routers in mixed vendor networks of up to one million routes to support Layer 3 VPNs.

Because using this statement can also enhance the Layer 3 VPN performance of Juniper Networks routers in networks where only Juniper Networks routers are deployed, we recommend configuring the statements in these networks as well.

---

To accept up to one million Layer 3 VPN route updates with unique inner VPN labels, configure the **l3vpn** statement. This statement is supported on indirectly connected PE routers only. Configuring this statement on a router that is directly connected to a PE router provides no benefit. You can configure the **l3vpn** statement on a router with a mix of links to both directly connected and indirectly connected PE routers.



**NOTE:** You cannot configure the **l3vpn** statement and sub-statements at same time that you have configured the **vpn-unequal-cost** statement.

To configure the router to accept up to one million Layer 3 VPN route updates with unique inner VPN labels:

1. Include the **l3vpn** statement.

```
[edit routing-options forwarding-table chained-composite-next-hop ingress]
user@host>set l3vpn
```

2. To enhance memory allocation to support a larger number of Layer 3 VPN labels, include the **vpn-label** statement.

```
[edit chassis memory-enhanced]
user@host>set vpn-label
```



**NOTE:** The **vpn-label** statement does not provide any functional changes when used on the MX Series routers. You can omit the configuration of this statement on MX Series routers.

For more information about configuring more memory for Layer 3 VPN labels, see the *Junos OS Administration Library for Routing Devices*.

After you have configured the **l3vpn** statement, you can determine whether or not a Layer 3 VPN route is a part of a composite next hop by examining the display output of the following commands:

- **show route route-value extensive**
- **show route forwarding-table destination destination-value extensive**

## Accepting More Than One Million Layer 3 VPN Route Updates

For Juniper Networks routers participating in a mixed vendor network with more than one million Layer 3 VPN labels, include the **extended-space** statement at the **[edit routing-options forwarding-table chained-composite-next-hop ingress l3vpn]** hierarchy level. The **extended-space** statement is disabled by default.



**BEST PRACTICE:** We recommend that you configure the **extended-space** statement in mixed vendor networks containing more than one million routes to support Layer 3 VPNs.

Because using this statements can also enhance the Layer 3 VPN performance of Juniper Networks routers in networks where only Juniper Networks routers are deployed, we recommend configuring the statement in these networks as well.

Using the **extended-space** statement can double the number of routes with unique inner VPN labels that can be processed by a Juniper Networks router. However, when configuring such very large-scale Layer 3 VPN scenarios, keep the following guidelines in mind:

- The **extended-space** statement is supported only on MX Series routers containing only MPCs.
- The chassis must be configured to use the **enhanced-ip** option in network services mode.

For more information about configuring chassis network services, see the *Junos OS Administration Library for Routing Devices*.

- Ensure that you configure per-packet load balancing for associated policies.

For more information about configuring policies, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices*.



**BEST PRACTICE:** We strongly recommend using 64-bit routing engines running 64-bit Junos OS to support Layer 3 VPN prefixes with unique inner VPN labels at higher scale.

To configure the router to accept more than one million Layer 3 VPN route updates with unique inner VPN labels:

1. Include the **l3vpn** statement.

```
[edit routing-options forwarding-table chained-composite-next-hop ingress]
user@host> set l3vpn
```

2. Include the **extended-space** statement.

```
[edit routing-options forwarding-table chained-composite-next-hop ingress l3vpn]
user@host> set extended-space
```

3. Configure chassis network services for enhanced mode.

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



**NOTE:** A router reboot might be required. See *Network Services Mode Overview* in the *Junos OS Administration Library for Routing Devices* for details.

After you have completed the configuration, you can determine whether or not a Layer 3 VPN route is a part of a composite next hop by examining the display output of the following commands:

- **show route *route-value* extensive**
- **show route forwarding-table destination *destination-value* extensive**

## Enabling Chained Composite Next Hops for IPv6 Labeled Unicast Routes

You can enable chained composite next hops for IPv6 labeled unicast routes by configuring the **labeled-bgp** and **inet6** statements:

- To enable chained composite next hops for inet6 labeled unicast routes, include the **inet6** statement at the **[edit routing-options forwarding-table chained-composite-next-hop ingress labeled-bgp]** hierarchy level. This statement is disabled by default.

### Related Documentation

- *Chained Composite Next Hops for Transit Devices*
- *Configuring the Junos OS to Allocate More Memory for Routing Tables, Firewall Filters, and Layer 3 VPN Labels*
- *Network Services Mode Overview*
- *Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers*



## CHAPTER 4

# Layer 3 VPN Configuration Examples

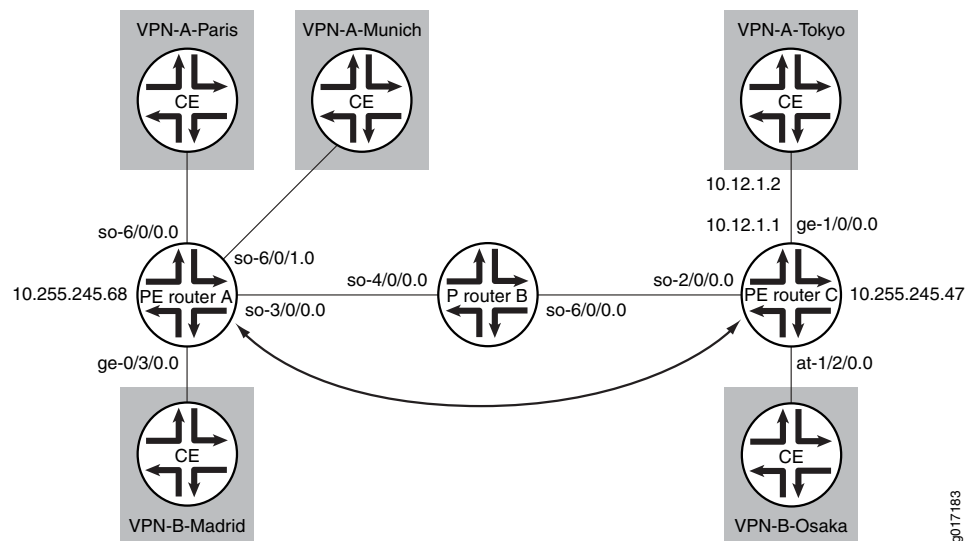
- [Configuring a Simple Full-Mesh VPN Topology on page 118](#)
- [Configuring a Full-Mesh VPN Topology with Route Reflectors on page 132](#)
- [Configuring Hub-and-Spoke VPN Topologies: One Interface on page 132](#)
- [Configuring Hub-and-Spoke VPN Topologies: Two Interfaces on page 144](#)
- [Configuring an LDP-over-RSVP VPN Topology on page 159](#)
- [Configuring an Application-Based Layer 3 VPN Topology on page 173](#)
- [Configuring an OSPF Domain ID for a Layer 3 VPN on page 178](#)
- [Configuring Overlapping VPNs Using Routing Table Groups on page 183](#)
- [Configuring Overlapping VPNs Using Automatic Route Export on page 194](#)
- [Configuring a GRE Tunnel Interface Between PE Routers on page 198](#)
- [Configuring a GRE Tunnel Interface Between a PE and CE Router on page 204](#)
- [Configuring an ES Tunnel Interface Between a PE and CE Router on page 208](#)
- [Layer 3 VPN Load Balancing Using IP Header Filtering on page 212](#)
- [Example: Disabling Normal TTL Decrementing in a VRF Routing Instance on page 227](#)
- [Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters on page 234](#)
- [Example: Configuring Route Resolution on PE Routers on page 238](#)
- [Example: Configuring Route Resolution on Route Reflectors on page 239](#)
- [Example: Configuring Link Protection with Host Fast Reroute on page 242](#)
- [Example: Configuring a Layer 3 VPN with Route Reflection and AS Override on page 258](#)
- [Example: Configuring MPLS Egress Protection for Layer 3 VPN Services on page 268](#)
- [Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP on page 283](#)
- [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 316](#)
- [Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths on page 332](#)
- [Example: Configuring Egress Protection for BGP Labeled Unicast on page 345](#)
- [Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains on page 358](#)

## Configuring a Simple Full-Mesh VPN Topology

This example shows how to set up a simple full-mesh service provider VPN configuration, which consists of the following components (see [Figure 20 on page 118](#)):

- Two separate VPNs (VPN-A and VPN-B)
- Two provider edge (PE) routers, both of which service VPN-A and VPN-B
- RSVP as the signaling protocol
- One RSVP label-switched path (LSP) that tunnels between the two PE routers through one provider (P) router

**Figure 20: Example of a Simple VPN Topology**



In this configuration, route distribution in VPN A from Router VPN-A-Paris to Router VPN-A-Tokyo occurs as follows:

1. The customer edge (CE) router VPN-A-Paris announces routes to the PE router Router A.
2. Router A installs the received announced routes into its VPN routing and forwarding (VRF) table, VPN-A.inet.0.
3. Router A creates an MPLS label for the interface between it and Router VPN-A-Paris.
4. Router A checks its VRF export policy.
5. Router A converts the Internet Protocol version 4 (IPv4) routes from Router VPN-A-Paris into VPN IPv4 format using its route distinguisher and announces these routes to PE Router C over the IBGP between the two PE routers.
6. Router C checks its VRF import policy and installs all routes that match the policy into its bgp.l3vpn.0 routing table. (Any routes that do not match are discarded.)

7. Router C checks its VRF import policy and installs all routes that match into its VPN-A.inet.0 routing table. The routes are installed in IPv4 format.
8. Router C announces its routes to the CE router Router VPN-A-Tokyo, which installs them into its master routing table. (For routing platforms running Junos OS, the master routing table is inet.0.)
9. Router C uses the LSP between it and Router A to route all packets from Router VPN-A-Tokyo that are destined for Router VPN-A-Paris.

The final section in this example consolidates the statements needed to configure VPN functionality on each of the service P routers shown in [Figure 20 on page 118](#).



**NOTE:** In this example, a private autonomous system (AS) number is used for the route distinguisher and the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

The following sections explain how to configure the VPN functionality on the PE and P routers. The CE routers have no information about the VPN, so you configure them normally.

- [Enabling an IGP on the PE and P Routers on page 119](#)
- [Enabling RSVP and MPLS on the P Router on page 119](#)
- [Configuring the MPLS LSP Tunnel Between the PE Routers on page 120](#)
- [Configuring IBGP on the PE Routers on page 121](#)
- [Configuring Routing Instances for VPNs on the PE Routers on page 122](#)
- [Configuring VPN Policy on the PE Routers on page 124](#)
- [Simple VPN Configuration Summarized by Router on page 127](#)

## Enabling an IGP on the PE and P Routers

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

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

## Enabling RSVP and MPLS on the P Router

On the P router, Router B, you must configure RSVP and MPLS because this router exists on the MPLS LSP path between the two PE routers, Router A and Router C:

```
[edit]
protocols {
  rsvp {
```

```
        interface so-4/0/0.0;
        interface so-6/0/0.0;
    }
    mpls {
        interface so-4/0/0.0;
        interface so-6/0/0.0;
    }
}
```

## Configuring the MPLS LSP Tunnel Between the PE Routers

In this configuration example, RSVP is used for VPN signaling. Therefore, in addition to configuring RSVP, you must enable traffic engineering support in an IGP and you must create an MPLS LSP to tunnel the VPN traffic.

On PE Router A, enable RSVP and configure one end of the MPLS LSP tunnel. In this example, traffic engineering support is enabled for OSPF. When configuring the MPLS LSP, include **interface** statements for all interfaces participating in MPLS, including the interfaces to the PE and CE routers. The statements for the interfaces between the PE and CE routers are needed so that the PE router can create an MPLS label for the private interface. In this example, the first **interface** statement configures MPLS on the interface connected to the LSP, and the remaining three configure MPLS on the interfaces that connect the PE router to the CE routers.

```
[edit]
protocols {
  rsvp {
    interface so-3/0/0.0;
  }
  mpls {
    label-switched-path RouterA-to-RouterC {
      to 10.255.245.47;
    }
    interface so-3/0/0.0;
    interface so-6/0/0.0;
    interface so-6/0/1.0;
    interface ge-0/3/0.0;
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-3/0/0.0;
    }
  }
}
```

On PE Router C, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces to the LSP and the CE routers.

```
[edit]
protocols {
  rsvp {
    interface so-2/0/0.0;
  }
}
```

```

mpls {
  label-switched-path RouterC-to-RouterA {
    to 10.255.245.68;
  }
  interface so-2/0/0.0;
  interface ge-1/0/0.0;
  interface at-1/2/0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface so-2/0/0.0;
  }
}
}

```

## Configuring IBGP on the PE Routers

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

- VPN family—To indicate that the IBGP session is for the VPN, include the **family inet-vpn** statement.
- Loopback address—Include the **local-address** statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the **lo0** interface at the **[edit interfaces]** hierarchy level. The example does not include this part of the router's configuration.
- Neighbor address—Include the **neighbor** statement, specifying the IP address of the neighboring PE router, which is its loopback (**lo0**) address.

On PE Router A, configure IBGP:

```

[edit]
protocols {
  bgp {
    group PE-RouterA-to-PE-RouterC {
      type internal;
      local-address 10.255.245.68;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.245.47;
    }
  }
}

```

On PE Router C, configure IBGP:

```

[edit]
protocols {
  bgp {
    group PE-RouterC-to-PE-RouterA {
      type internal;
      local-address 10.255.245.47;
      family inet-vpn {

```

```

        unicast;
      }
      neighbor 10.255.245.68;
    }
  }
}

```

## Configuring Routing Instances for VPNs on the PE Routers

Both PE routers service VPN-A and VPN-B, so you must configure two routing instances on each router, one for each VPN. For each VPN, you must define the following in the routing instance:

- Route distinguisher, which must be unique for each routing instance on the PE router.
- It is used to distinguish the addresses in one VPN from those in another VPN.
- Instance type of **vrf**, which creates the VRF table on the PE router.
- Interfaces connected to the CE routers.
- VRF import and export policies, which must be the same on each PE router that services the same VPN. Unless an import policy contains only a **then reject** statement, it must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails.



**NOTE:** In this example, a private AS number is used for the route distinguisher. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

On PE Router A, configure the following routing instance for VPN-A. In this example, Router A uses static routes to distribute routes to and from the two CE routers to which it is connected.

```

[edit]
routing-instance {
  VPN-A-Paris-Munich {
    instance-type vrf;
    interface so-6/0/0.0;
    interface so-6/0/1.0;
    route-distinguisher 65535:0;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    routing-options {
      static {
        route 172.16.0.0/16 next-hop so-6/0/0.0;
        route 172.17.0.0/16 next-hop so-6/0/1.0;
      }
    }
  }
}

```

```
}

```

On PE Router C, configure the following routing instance for VPN-A. In this example, Router C uses BGP to distribute routes to and from the CE router to which it is connected.

```
[edit]
routing-instance {
  VPN-A-Tokyo {
    instance-type vrf;
    interface ge-1/0/0.0;
    route-distinguisher 65535:1;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    protocols {
      bgp {
        group VPN-A-Site2 {
          peer-as 1;
          neighbor 10.12.1.2;
        }
      }
    }
  }
}
```

On PE Router A, configure the following routing instance for VPN-B. In this example, Router A uses OSPF to distribute routes to and from the CE router to which it is connected.

```
[edit]
policy-options {
  policy-statement bgp-to-ospf {
    from {
      protocol bgp;
      route-filter 192.168.1.0/24 orlonger;
    }
    then accept;
  }
}
routing-instance {
  VPN-B-Madrid {
    instance-type vrf;
    interface ge-0/3/0.0;
    route-distinguisher 65535:2;
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
    protocols {
      ospf {
        export bgp-to-ospf;
        area 0.0.0.0 {
          interface ge-0/3/0;
        }
      }
    }
  }
}
```

On PE Router C, configure the following routing instance for VPN-B. In this example, Router C uses RIP to distribute routes to and from the CE router to which it is connected.

```
[edit]
policy-options {
  policy-statement bgp-to-rip {
    from {
      protocol bgp;
      route-filter 192.168.2.0/24 orlonger;
    }
    then accept;
  }
}
routing-instance {
  VPN-B-Osaka {
    instance-type vrf;
    interface at-1/2/0.0;
    route-distinguisher 65535:3;
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
    protocols {
      rip {
        group PE-C-to-VPN-B {
          export bgp-to-rip;
          neighbor at-1/2/0;
        }
      }
    }
  }
}
```

## Configuring VPN Policy on the PE Routers

Configure the VPN import and export policies on each PE router so that the appropriate routes are installed in the PE router's VRF tables. The VRF table is used to forward packets within a VPN. For VPN-A, the VRF table is VPN-A.inet.0, and for VPN-B it is VPN-B.inet.0.

In the VPN policy, you also configure VPN target communities.

In the following example, a private AS number is used for the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number. The policy qualifiers shown in this example are only those needed for the VPN to function. You can configure additional qualifiers, as needed, for any policies that you configure.

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

```
[edit]
policy-options {
  policy-statement VPN-A-import {
    term a {
      from {
        protocol bgp;
        community VPN-A;
      }
      then accept;
    }
    term b {
```

```

        then reject;
    }
}
policy-statement VPN-A-export {
    term a {
        from protocol static;
        then {
            community add VPN-A;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-import {
    term a {
        from {
            protocol bgp;
            community VPN-B;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-export {
    term a {
        from protocol ospf;
        then {
            community add VPN-B;
            accept;
        }
    }
    term b {
        then reject;
    }
}
community VPN-A members target:65535:4;
community VPN-B members target:65535:5;
}

```

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

```

[edit]
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {

```

```
        then reject;
    }
}
policy-statement VPN-A-export {
    term a {
        from protocol bgp;
        then {
            community add VPN-A;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-import {
    term a {
        from {
            protocol bgp;
            community VPN-B;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-export {
    term a {
        from protocol rip;
        then {
            community add VPN-B;
            accept;
        }
    }
    term b {
        then reject;
    }
}
community VPN-A members target:65535:4;
community VPN-B members target:65535:5;
}
```

To apply the VPN policies on the routers, include the **vrf-export** and **vrf-import** statements when you configure the routing instance. For both VPNs, the VRF import and export policies handle the route distribution across the IBGP session running between the PE routers.

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

```
[edit]
routing-instance {
    VPN-A-Paris-Munich {
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
}
```

```

VPN-B-Madrid {
  vrf-import VPN-B-import;
  vrf-export VPN-B-export;
}

```

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

```

[edit]
routing-instance {
  VPN-A-Tokyo {
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
  }
  VPN-B-Osaka {
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
  }
}

```

## Simple VPN Configuration Summarized by Router

### Router A (PE Router)

Routing Instance for VPN-A	<pre> routing-instance {   VPN-A-Paris-Munich {     instance-type vrf;     interface so-6/0/0.0;     interface so-6/0/1.0;     route-distinguisher 65535:0;     vrf-import VPN-A-import;     vrf-export VPN-A-export;   } } </pre>
Instance Routing Protocol	<pre> routing-options {   static {     route 172.16.0.0/16 next-hop so-6/0/0.0;     route 172.17.0.0/16 next-hop so-6/0/1.0;   } } </pre>
Routing Instance for VPN-B	<pre> routing-instance {   VPN-B-Madrid {     instance-type vrf;     interface ge-0/3/0.0;     route-distinguisher 65535:2;     vrf-import VPN-B-import;     vrf-export VPN-B-export;   } } </pre>
Instance Routing Protocol	<pre> protocols {   ospf {     area 0.0.0.0 {       interface ge-0/3/0;     }   } } </pre>

	<pre>         }       }     } </pre>
<b>Master Protocol Instance</b>	<pre> protocols { } </pre>
<b>Enable RSVP</b>	<pre> rsvp {   interface so-3/0/0.0; } </pre>
<b>Configure an MPLS LSP</b>	<pre> mpls {   label-switched-path RouterA-to-RouterC {     to 10.255.245.47;   }   interface so-3/0/0.0;   interface so-6/0/0.0;   interface so-6/0/1.0;   interface ge-0/3/0.0; } </pre>
<b>Configure IBGP</b>	<pre> bgp {   group PE-RouterA-to-PE-RouterC {     type internal;     local-address 10.255.245.68;     family inet-vpn {       unicast;     }     neighbor 10.255.245.47;   } } </pre>
<b>Configure OSPF for Traffic Engineering Support</b>	<pre> ospf {   traffic-engineering;   area 0.0.0.0 {     interface so-3/0/0.0;   } } </pre>
<b>Configure VPN Policy</b>	<pre> policy-options {   policy-statement VPN-A-import {     term a {       from {         protocol bgp;         community VPN-A;       }       then accept;     }     term b {       then reject;     }   }   policy-statement VPN-A-export {     term a {       from protocol static;     }   } } </pre>

```

        then {
            community add VPN-A;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-import {
    term a {
        from {
            protocol bgp;
            community VPN-B;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-export {
    term a {
        from protocol ospf;
        then {
            community add VPN-B;
            accept;
        }
    }
    term b {
        then reject;
    }
}
community VPN-A members target:65535:4;
community VPN-B members target:65535:5;
}

```

#### Router B (P Router)

<b>Master Protocol Instance</b>	<pre> protocols { } </pre>
<b>Enable RSVP</b>	<pre> rsvp {     interface so-4/0/0.0;     interface so-6/0/0.0; } </pre>
<b>Enable MPLS</b>	<pre> mpls {     interface so-4/0/0.0;     interface so-6/0/0.0; } </pre>

### Router C (PE Router)

Routing Instance for VPN-A	<pre> routing-instance {   VPN-A-Tokyo {     instance-type vrf;     interface ge-1/0/0.0;     route-distinguisher 65535:1;     vrf-import VPN-A-import;     vrf-export VPN-A-export;   } } </pre>
Instance Routing Protocol	<pre> protocols {   bgp {     group VPN-A-Site2 {       peer-as 1;       neighbor 10.12.1.2;     }   } } </pre>
Routing Instance for VPN-B	<pre> VPN-B-Osaka {   instance-type vrf;   interface at-1/2/0.0;   route-distinguisher 65535:3;   vrf-import VPN-B-import;   vrf-export VPN-B-export; } </pre>
Instance Routing Protocol	<pre> protocols {   rip {     group PE-C-to-VPN-B {       neighbor at-1/2/0;     }   } } </pre>
Master Protocol Instance	<pre> protocols { } </pre>
Enable RSVP	<pre> rsvp {   interface so-2/0/0.0; } </pre>
Configure an MPLS LSP	<pre> mpls {   label-switched-path RouterC-to-RouterA {     to 10.255.245.68;   }   interface so-2/0/0.0;   interface ge-1/0/0.0;   interface at-1/2/0.0; } </pre>
Configure IBGP	<pre> bgp { </pre>

	<pre> group PE-RouterC-to-PE-RouterA {   type internal;   local-address 10.255.245.47;   family inet-vpn {     unicast;   }   neighbor 10.255.245.68; } </pre>
<b>Configure OSPF for Traffic Engineering Support</b>	<pre> ospf {   traffic-engineering;   area 0.0.0.0 {     interface so-2/0/0.0;   } } </pre>
<b>Configure VPN Policy</b>	<pre> policy-options {   policy-statement VPN-A-import {     term a {       from {         protocol bgp;         community VPN-A;       }       then accept;     }     term b {       then reject;     }   }   policy-statement VPN-A-export {     term a {       from protocol bgp;       then {         community add VPN-A;         accept;       }     }     term b {       then reject;     }   }   policy-statement VPN-B-import {     term a {       from {         protocol bgp;         community VPN-B;       }       then accept;     }     term b {       then reject;     }   }   policy-statement VPN-B-export { </pre>

```
term a {  
  from protocol rip;  
  then {  
    community add VPN-B;  
    accept;  
  }  
}  
term b {  
  then reject;  
}  
}  
community VPN-A members target:65535:4;  
community VPN-B members target:65535:5;  
}
```

---

## Configuring a Full-Mesh VPN Topology with Route Reflectors

This example is a variation of the full-mesh VPN topology example (described in [“Configuring a Simple Full-Mesh VPN Topology” on page 118](#)) in which one of the PE routers is a BGP route reflector. In this variation, Router C in [Figure 20 on page 118](#) is a route reflector. The only change to its configuration is that you need to include the **cluster** statement when configuring the BGP group:

```
[edit]  
protocols {  
  bgp {  
    group PE-RouterC-to-PE-RouterA {  
      type internal;  
      local-address 10.255.245.47;  
      family inet-vpn {  
        unicast;  
      }  
      neighbor 10.255.245.68;  
      cluster 4.3.2.1;  
    }  
  }  
}
```

For the complete configuration example of Router C, see [“Configuring a Full-Mesh VPN Topology with Route Reflectors” on page 132](#).

---

## Configuring Hub-and-Spoke VPN Topologies: One Interface

Use a one-interface configuration to advertise a default route from a hub or hubs.

Figure 21: Example of a Hub-and-Spoke VPN Topology with One Interface

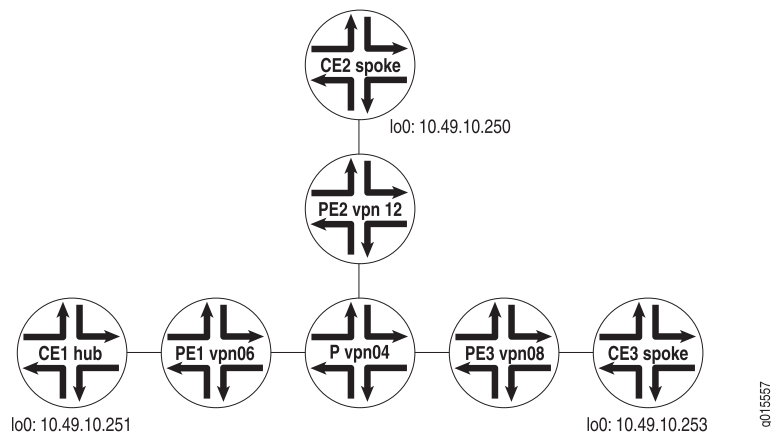


Figure 21 on page 133 illustrates a Layer 3 VPN hub-and-spoke application where there is only one interface between the hub CE (CE1) and the hub PE (PE1). This is the recommended way of configuring hub-and-spoke topologies.

In this configuration, a default route is advertised from the hub to the spokes. If more specific spoke CE routes need to be exchanged between spoke CE routers, then two interfaces are needed between the hub CE and hub PE. See [“Configuring Hub-and-Spoke VPN Topologies: Two Interfaces” on page 144](#) for a two-interface example.

In this configuration example, spoke route distribution is as follows:

1. Spoke CE2 advertises its routes to spoke PE2.
2. Spoke PE2 installs routes from CE2 into its VPN routing and forwarding (VRF) table.
3. Spoke PE2 checks its VRF export policy, adds the route target community, and announces the routes to hub PE1.
4. Hub PE1 checks its VRF import policy and installs routes that match the import policy into table `bgp.l3vpn.0`.
5. Hub PE1 installs routes from table `bgp.l3vpn.0` into the hub VRF table.
6. Hub PE1 announces routes from the hub VRF table to the hub CE1.

In this configuration example, default route distribution is as follows:

1. Hub CE1 announces a default route to hub PE1.
2. Hub PE1 installs the default route into the hub VRF table.
3. Hub PE1 checks its VRF export policy, adds the route target community and announces the default route to spoke PE2 and PE3.
4. Spoke PE2 and PE3 check their VRF import policy and install the default route into table `bgp.l3vpn.0`.

5. Spoke PE2 and PE3 install the routes from table `bgp.l3vpn.0` into their spoke VRF tables.
6. Spoke PE2 and PE3 announce the default route from the spoke VRF table to spoke CE2 and CE3.

The following sections describe how to configure a hub-and-spoke topology with one interface based on the topology illustrated in [Figure 21 on page 133](#):

- [Configuring Hub CE1 on page 134](#)
- [Configuring Hub PE1 on page 135](#)
- [Configuring the P Router on page 135](#)
- [Configuring Spoke PE2 on page 136](#)
- [Configuring Spoke PE3 on page 137](#)
- [Configuring Spoke CE2 on page 138](#)
- [Configuring Spoke CE3 on page 139](#)
- [Enabling Egress Features on the Hub PE Router on page 141](#)

## Configuring Hub CE1

Configure hub CE1 as follows:

```
[edit routing-options]
static {
    route 0.0.0.0/0 discard;
}
autonomous-system 100;
[edit protocols]
bgp {
    group hub {
        type external;
        export default;
        peer-as 200;
        neighbor 10.49.4.1;
    }
}
[edit policy-statement]
default {
    term 1 {
        from {
            protocol static;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

## Configuring Hub PE1

Configure hub PE1 as follows:

```
[edit]
routing-instances {
  hub {
    instance-type vrf;
    interface t3-0/0/0 {
      encapsulation frame-relay;
      unit 0 {
        dlci 16;
        family inet {
          address 10.49.4.1/30;
        }
      }
    }
    vrf-target {
      import target:200:100;
      export target:200:101;
    }
    protocols {
      bgp {
        group hub {
          type external;
          peer-as 100;
          as-override;
          neighbor 10.49.4.2;
        }
      }
    }
  }
}
```

## Configuring the P Router

Configure the P Router as follows:

```
[edit]
interfaces {
  t3-0/1/1 {
    unit 0 {
      family inet {
        address 10.49.2.1/30;
      }
      family mpls;
    }
  }
  t3-0/1/3 {
    unit 0 {
      family inet {
        address 10.49.0.2/30;
      }
      family mpls;
    }
  }
}
```

```

    }
    t1-0/2/0 {
        unit 0 {
            family inet {
                address 10.49.1.2/30;
            }
            family mpls;
        }
    }
}
[edit]
protocols {
    ospf {
        area 0.0.0.0 {
            interface t3-0/1/3.0;
            interface t1-0/2/0.0;
            interface t3-0/1/1.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}
ldp {
    interface t3-0/1/1.0;
    interface t3-0/1/3.0;
    interface t1-0/2/0.0;
}
}

```

## Configuring Spoke PE2

Configure spoke PE2 as follows:

```

[edit]
interfaces {
    t3-0/0/0 {
        unit 0 {
            family inet {
                address 10.49.0.1/30;
            }
            family mpls;
        }
    }
    t1-0/1/2 {
        unit 0 {
            family inet {
                address 10.49.3.1/30;
            }
        }
    }
}
[edit protocols]
bgp {
    group ibgp {
        type internal;
        local-address 10.255.14.182;
    }
}

```

```

    peer-as 200;
    neighbor 10.255.14.176 {
        family inet-vpn {
            unicast;
        }
    }
}
}
ospf {
    area 0.0.0.0 {
        interface t3-0/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface t3-0/0/0.0;
}
[edit]
routing-instances {
    spoke {
        instance-type vrf;
        interface t1-0/1/2.0;
        vrf-target {
            import target:200:101;
            export target:200:100;
        }
        protocols {
            bgp {
                group spoke {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.3.2;
                }
            }
        }
    }
}
}

```

## Configuring Spoke PE3

Configure spoke PE3 as follows:

```

[edit]
interfaces {
    t3-0/0/0 {
        unit 0 {
            family inet {
                address 10.49.6.1/30;
            }
        }
    }
    t3-0/0/1 {
        unit 0 {

```

```

        family inet {
            address 10.49.2.2/30;
        }
        family mpls;
    }
}
[edit protocols]
bgp {
    group ibgp {
        type internal;
        local-address 10.255.14.178;
        peer-as 200;
        neighbor 10.255.14.176 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface t3-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface t3-0/0/1.0;
}
[edit]
routing-instances {
    spoke {
        instance-type vrf;
        interface t3-0/0/0.0;
        vrf-target {
            import target:200:101;
            export target:200:100;
        }
        protocols {
            bgp {
                group spoke {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.6.2;
                }
            }
        }
    }
}

```

## Configuring Spoke CE2

Configure spoke CE2 as follows:

```
[edit routing-options]
autonomous-system 100;
{edit protocols}
bgp {
  group spoke {
    type external;
    export loopback;
    peer-as 200;
    neighbor 10.49.3.1;
  }
}
```

## Configuring Spoke CE3

Configure spoke CE3 as follows:

```
[edit routing-options]
autonomous-system 100;
[edit protocols]
bgp {
  group spoke {
    type external;
    export loopback;
    peer-as 200;
    neighbor 10.49.6.1;
  }
}
```

In this configuration example, traffic forwarding is as follows between spoke CE2 and hub CE1:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```
0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                   AS path: 200 200 I
                   > to 10.49.3.1 via t1-3/0/1.0
```

2. Spoke PE2 performs a route lookup in the spoke VRF table and forwards the traffic to hub PE2 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```
0.0.0.0/0          *[BGP/170] 01:35:45, localpref 100, from 10.255.14.176
                   AS path: 100 I
                   > via t3-0/0/1.0, Push 100336, Push 100224(top)
```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label **100336**.

```
100336             *[VPN/170] 01:37:03
                   > to 10.49.4.2 via t3-0/0/0.0, Pop
```

4. Hub PE1 forwards the traffic out the interface **t3-0/0/0.0** to hub CE1.

In this configuration example, traffic forwarding is as follows between hub CE1 and spoke CE2:

1. Hub CE1 forwards traffic to the hub PE1 using the route learned through BGP.

```

10.49.10.250/32    *[BGP/170] 02:28:46, localpref 100
                  AS path: 200 200 I
                  > to 10.49.4.1 via t3-3/1/0.0

```

2. Hub PE1 does a route lookup in the hub VRF table and forwards the traffic to spoke PE2 (through the P router—PE1 pushes two labels).

```

10.49.10.250/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.182
                  AS path: 100 I
                  > via t1-0/1/0.0, Push 100352, Push 100208(top)

```

3. Spoke PE2 does a route lookup in the mpls.0 table for the VPN label **100352**.

```

100352             *[VPN/170] 02:31:39
                  > to 10.49.3.2 via t1-0/1/2.0, Pop

```

4. Spoke PE2 forwards the traffic out the interface **t1-0/1/2.0** to spoke CE2.

In this configuration example, traffic forwarding is as follows between spoke CE2 and spoke CE3:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```

0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                  AS path: 200 200 I
                  > to 10.49.3.1 via t1-3/0/1.0

```

2. Spoke PE2 does a route lookup in the spoke VRF table and forwards the traffic to hub PE1 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```

0.0.0.0/0          *[BGP/170] 01:35:45, localpref 100, from 10.255.14.176
                  AS path: 100 I
                  > via t3-0/0/1.0, Push 100336, Push 100224(top)

```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label **100336**.

```

100336             *[VPN/170] 01:37:03
                  > to 10.49.4.2 via t3-0/0/0.0, Pop

```

4. Hub PE1 forwards the traffic out the interface **t3-0/0/0.0** to the hub CE1.
5. Hub CE1 forwards the traffic to hub PE1 using the router learned through BGP.

```

10.49.10.253/32    *[BGP/170] 02:40:03, localpref 100
                  AS path: 200 200 I
                  > to 10.49.4.1 via t3-3/1/0.0

```

6. Hub PE1 does a route lookup in the hub VRF table and forwards the traffic to spoke PE3 (through the P router—PE1 pushes two labels).

```

10.49.10.253/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.178
                  AS path: 100 I
                  > via t1-0/1/0.0, Push 100128, Push 100192(top)

```

7. Spoke PE3 does a route lookup in the mpls.0 table for VPN label **100128**.

```

100128                *[VPN/170] 02:41:30
                      > to 10.49.6.2 via t3-0/0/0.0, Pop

```

8. Spoke PE3 forwards the traffic out the interface **t3-0/0/0.0** to spoke CE3.

If egress features are needed on the hub PE that require an IP forwarding lookup on the hub VRF routing table, see [“Enabling Egress Features on the Hub PE Router” on page 141](#).

## Enabling Egress Features on the Hub PE Router

This example is provided in conjunction with [“Configuring Hub-and-Spoke VPN Topologies: One Interface” on page 132](#). This example also uses the topology illustrated in [Figure 21 on page 133](#).

If egress features are needed on the hub PE that require an IP forwarding lookup on the hub VRF routing table, the configuration detailed in [“Configuring Hub-and-Spoke VPN Topologies: One Interface” on page 132](#) will not work. Applying the **vrf-table-label** statement on the hub routing instance forces traffic from a remote spoke PE to be forwarded to the hub PE and forces an IP lookup to be performed. Because specific spoke routes are in the hub VRF table, traffic will be forwarded to a spoke PE without going through the hub CE.

The hub PE advertises the default route as follows, using VPN label 1028:

```

hub.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
* 0.0.0.0/0 (1 entry, 1 announced)
  BGP group ibgp type Internal
    Route Distinguisher: 10.255.14.176:2
    VPN Label: 1028
    Nexthop: Self
    Localpref: 100
    AS path: 100 I
    Communities: target:200:101

```

Incoming traffic is forwarded using VPN label 1028. The mpls.0 table shows that an IP lookup in the table hub.inet.0 is required:

```

1028                *[VPN/0] 00:00:27
                      to table hub.inet.0, Pop

```

However, the hub VRF table hub.inet.0 contains specific spoke routes:

```

10.49.10.250/32      *[BGP/170] 00:00:05, localpref 100, from 10.255.14.182
                      AS path: 100 I
                      > via t1-0/1/0.0, Push 100352, Push 100208(top)
10.49.10.253/32      *[BGP/170] 00:00:05, localpref 100, from 10.255.14.178
                      AS path: 100 I
                      > via t1-0/1/0.0, Push 100128, Push 100192(top)

```

Because of this, traffic is forwarded directly to the spoke PEs without going through the hub CE. To prevent this, you must configure a secondary routing instance for downstream traffic in the hub PE1.

## Configuring Hub PE1

Configure hub PE1 as follows:

[edit]

```
routing-instances {
  hub {
    instance-type vrf;
    interface t3-0/0/0.0;
    vrf-target {
      import target:200:100;
      export target:200:101;
    }
    no-vrf-advertise;
    routing-options {
      auto-export;
    }
    protocols {
      bgp {
        group hub {
          type external;
          peer-as 100;
          as-override;
          neighbor 10.49.4.2;
        }
      }
    }
  }
  hub-downstream {
    instance-type vrf;
    vrf-target target:200:101;
    vrf-table-label;
    routing-options {
      auto-export;
    }
  }
}
```

When the **no-vrf-advertise** statement is used at the **[edit routing-instances hub]** hierarchy level, no routing table groups or VRF export policies are required. The **no-vrf-advertise** statement configures the hub PE not to advertise VPN routes from the primary routing-instance **hub**. These routes are instead advertised from the secondary routing instance **hub\_downstream**. See the routing instances configuration guidelines in the *Junos OS Routing Protocols Library for Routing Devices* for more information about the **no-vrf-advertise** statement.

The **auto-export** statement at the **[edit routing-instances hub-downstream routing-options]** hierarchy level identifies routes exported from the hub instance to the hub-downstream instance by looking at the route targets defined for each routing instance. See the routing instances configuration guidelines in the *Junos OS Routing Protocols Library for Routing Devices* for more information about using the **auto-export** statement. See *Configuring Overlapping VPNs Using Automatic Route Export* for more examples of export policy.

With this configuration on hub PE, spoke-to-spoke CE traffic goes through the hub CE and permits egress features (such as filtering) to be enabled on the hub PE.

In this configuration example, traffic forwarding is as follows between spoke CE2 and spoke CE3:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```
0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                   AS path: 200 200 I
                   > to 10.49.3.1 via t1-3/0/1.0
```

2. Spoke PE2 does a route lookup in the spoke VRF table and forwards the traffic to hub PE1 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```
spoke.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:00:09, localpref 100, from 10.255.14.176
                   AS path: 100 I
                   > via t3-0/0/0.0, Push 1029, Push 100224(top)
```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label 1029.

```
mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
1029               *[VPN/0] 00:11:49
                   to table hub_downstream.inet.0, Pop
```

The VPN label 1029 is advertised because:

- a. The **vrf-table-label** statement is applied at the **[edit routing-instances hub\_downstream]** hierarchy level in the hub PE1 configuration.
- b. The **no-vrf-advertise** statement is applied at the **[edit routing-instances hub]** hierarchy level, instructing the router to advertise the route from the secondary table.

Therefore, IP lookups are performed in the **hub\_downstream.inet.0** table, not in the **hub.inet.0** table.

Issue the **show route advertising-protocol** command on the hub PE to a spoke PE to verify the VPN label 1029 advertisement:

```
user@host> show route advertising-protocol
hub_downstream.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
* 0.0.0.0/0 (1 entry, 1 announced)
  BGP group ibgp type Internal
  Route Distinguisher: 10.255.14.176:3
  VPN Label: 1029
  Nexthop: Self
  Localpref: 100
  AS path: 100 I
  Communities: target:200:101
```

4. Hub PE1 performs an IP lookup in the **hub\_downstream.inet.0** table and forwards the traffic out interface **t3-0/0/0.0** to hub CE1.

```
hub_downstream.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
```

```

0.0.0.0/0 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Next-hop reference count: 4
            Source: 10.49.4.2
            Next hop: 10.49.4.2 via t3-0/0/0.0, selected
            State: <Secondary Active Ext>
            Peer AS: 100
            Age: 3:03
            Task: BGP_100.10.49.4.2+1707
            Announcement bits (2): 0-KRT 2-BGP.0.0.0.0+179
            AS path: 100 I
            Communities: target:200:101
            Localpref: 100
            Router ID: 10.49.10.251
            Primary Routing Table hub.inet.0

```

The primary routing table is **hub.inet.0**, indicating that this route was exported from table **hub.inet.0** into this **hub\_downstream.inet.0** table as a result of the **no-vrf-advertise** statement at the **[edit routing-instances hub]** hierarchy level and the **auto-export** statement at the **[edit routing-instances hub-downstream routing-options]** hierarchy level in the hub PE1 configuration.

5. Hub CE1 forwards the traffic back to hub PE1 using the router learned through BGP.

```

10.49.10.253/32    *[BGP/170] 02:40:03, localpref 100
                  AS path: 200 200 I
                  > to 10.49.4.1 via t3-3/1/0.0

```

6. Hub PE1 performs a route lookup in the hub VRF table and forwards the traffic to spoke PE3 (through the P router—PE1 pushes two labels).

```

10.49.10.253/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.178
                  AS path: 100 I
                  > via t1-0/1/0.0, Push 100128, Push 100192(top)

```

7. Spoke PE3 performs a route lookup in the mpls.0 table for VPN label **100128**.

```

100128            *[VPN/170] 02:41:30
                  > to 10.49.6.2 via t3-0/0/0.0, Pop

```

8. Spoke PE3 forwards traffic out interface **t3-0/0/0.0** to spoke CE3.

## Configuring Hub-and-Spoke VPN Topologies: Two Interfaces

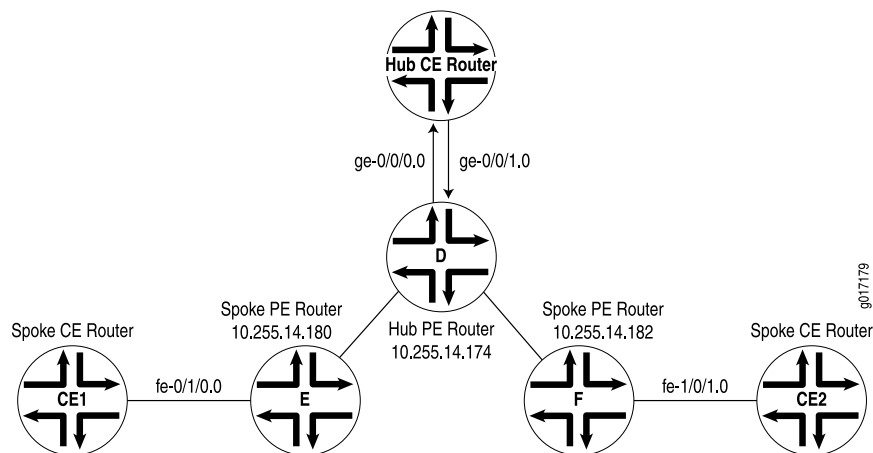
Use a two-interface configuration to propagate routes from spoke to spoke.

The example in this section configures a hub-and-spoke topology with two interfaces using the following components (see [Figure 22 on page 145](#)):

- One hub PE router (Router D).
- One hub CE router connected to the hub PE router. For this hub-and-spoke VPN topology to function properly, there must be two interfaces connecting the hub PE router to the hub CE router, and each interface must have its own VRF table on the PE router:

- The first interface (here, interface ge-0/0/0.0) is used to announce spoke routes to the hub CE router. The VRF table associated with this interface contains the routes being announced by the spoke PE routers to the hub CE router.
- The second interface (here, interface ge-0/0/1.0) is used to receive route announcements from the hub CE that are destined for the hub-and-spoke routers. The VRF table associated with this interface contains the routes announced by the hub CE router to the spoke PE routers. For this example, two separate physical interfaces are used. It would also work if you were to configure two separate logical interfaces sharing the same physical interface between the hub PE router and the hub CE router.
- Two spoke PE routers (Router E and Router F).
- Two spoke CE routers (CE1 and CE2), one connected to each spoke PE router.
- LDP as the signaling protocol.

**Figure 22: Example of a Hub-and-Spoke VPN Topology with Two Interfaces**



In this configuration, route distribution from spoke CE Router CE1 occurs as follows:

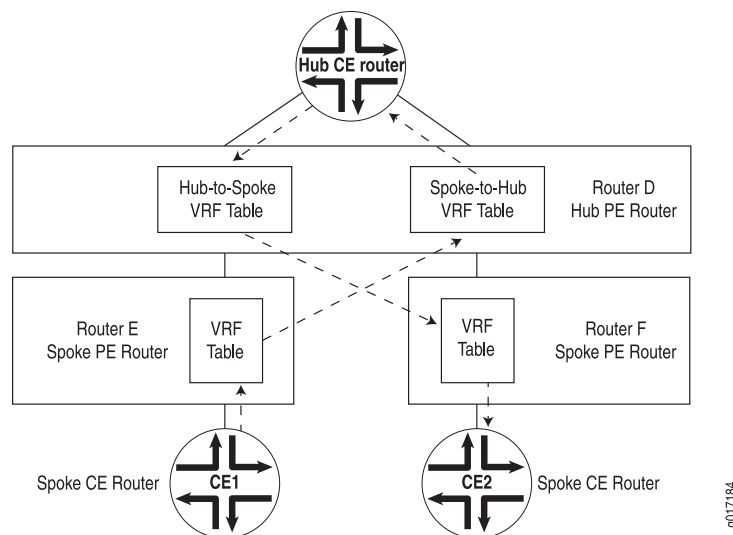
1. Spoke Router CE1 announces its routes to spoke PE Router E.
2. Router E installs the routes from CE1 into its VRF table.
3. After checking its VRF export policy, Router E adds the spoke target community to the routes from Router CE1 that passed the policy and announces them to the hub PE router, Router D.
4. Router D checks the VRF import policy associated with interface ge-0/0/0.0 and places all routes from spoke PE routers that match the policy into its bgp.l3vpn routing table. (Any routes that do not match are discarded.)
5. Router D checks its VRF import policy associated with interface ge-0/0/0.0 and installs all routes that match into its spoke VRF table. The routes are installed with the spoke target community.
6. Router D announces routes to the hub CE over interface ge-0/0/0.

7. The hub CE router announces the routes back to the hub PE Router D over the second interface to the hub router, interface ge-0/0/1.
8. The hub PE router installs the routes learned from the hub CE router into its hub VRF table, which is associated with interface ge-0/0/1.
9. The hub PE router checks the VRF export policy associated with interface ge-0/0/1.0 and announces all routes that match to all spokes after adding the hub target community.

Figure 23 on page 146 illustrates how routes are distributed from this spoke router to the other spoke CE router, Router CE2. The same path is followed if you issue a **traceroute** command from Router CE1 to Router CE2.

The final section in this example, “[Hub-and-Spoke VPN Configuration Summarized by Router](#)” on page 154, consolidates the statements needed to configure VPN functionality for each of the service provider routers shown in Figure 22 on page 145.

**Figure 23: Route Distribution Between Two Spoke Routers**



The following sections explain how to configure the VPN functionality for a hub-and-spoke topology on the hub-and-spoke PE routers. The CE routers do not have any information about the VPN, so you configure them normally.

- [Enabling an IGP on the Hub-and-Spoke PE Routers on page 147](#)
- [Configuring LDP on the Hub-and-Spoke PE Routers on page 147](#)
- [Configuring IBGP on the PE Routers on page 148](#)
- [Configuring VPN Routing Instances on the Hub-and-Spoke PE Routers on page 149](#)
- [Configuring VPN Policy on the PE Routers on page 151](#)
- [Hub-and-Spoke VPN Configuration Summarized by Router on page 154](#)

## Enabling an IGP on the Hub-and-Spoke PE Routers

To allow the hub-and-spoke PE routers to exchange routing information, you must configure an IGP on all these routers or you must configure static routes. You configure the IGP on the master instance of the routing protocol process (rpd) (that is, at the **[edit protocols]** hierarchy level), not within the routing instance (that is, not at the **[edit routing-instances]** hierarchy level).

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

In the route distribution in a hub-and-spoke topology, if the protocol used between the CE and PE routers at the hub site is BGP, the hub CE router announces all routes received from the hub PE router and the spoke routers back to the hub PE router and all the spoke routers. This means that the hub-and-spoke PE routers receive routes that contain their AS number. Normally, when a route contains this information, it indicates that a routing loop has occurred and the router rejects the routes. However, for the VPN configuration to work, the hub PE router and the spoke routers must accept these routes. To enable this, include the **loops** option when configuring the AS at the **[edit routing-options]** hierarchy level on the hub PE router and all the spoke routers. For this example configuration, you specify a value of 1. You can specify a number from 0 through 10.

```
[edit routing-options]
autonomous-system as-number loops 1;
```

## Configuring LDP on the Hub-and-Spoke PE Routers

Configure LDP on the interfaces between the hub-and-spoke PE routers that participate in the VPN.

On hub PE Router D, configure LDP:

```
[edit protocols]
ldp {
  interface so-1/0/0.0;
  interface t3-1/1/0.0;
}
```

On spoke PE Router E, configure LDP:

```
[edit protocols]
ldp {
  interface fe-0/1/2.0;
}
```

On spoke PE router Router F, configure LDP:

```
[edit protocols]
ldp {
  interface fe-1/0/0.0;
}
```

## Configuring IBGP on the PE Routers

On the hub-and-spoke PE routers, configure an IBGP session with the following properties:

- VPN family—To indicate that the IBGP session is for the VPN, include the **family inet-vpn** statement.
- Loopback address—Include the **local-address** statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the **lo0** interface at the **[edit interfaces]** hierarchy level. The example does not include this part of the router's configuration.
- Neighbor address—Include the **neighbor** statement. On the hub router, specify the IP address of each spoke PE router, and on the spoke router, specify the address of the hub PE router.

For the hub router, you configure an IBGP session with each spoke, and for each spoke router, you configure an IBGP session with the hub. There are no IBGP sessions between the two spoke routers.

On hub Router D, configure IBGP. The first **neighbor** statement configures an IBGP session to spoke Router E, and the second configures a session to spoke Router F.

```
[edit protocols]
bgp {
  group Hub-to-Spokes {
    type internal;
    local-address 10.255.14.174;
    family inet-vpn {
      unicast;
    }
    neighbor 10.255.14.180;
    neighbor 10.255.14.182;
  }
}
```

On spoke Router E, configure an IBGP session to the hub router:

```
[edit protocols]
bgp {
  group Spoke-E-to-Hub {
    type internal;
    local-address 10.255.14.180;
    neighbor 10.255.14.174 {
      family inet-vpn {
        unicast;
      }
    }
  }
}
```

On spoke Router F, configure an IBGP session to the hub router:

```
[edit protocols]
bgp {
```

```

group Spoke-F-to-Hub {
  type internal;
  local-address 10.255.14.182;
  neighbor 10.255.14.174 {
    family inet-vpn {
      unicast;
    }
  }
}

```

## Configuring VPN Routing Instances on the Hub-and-Spoke PE Routers

For the hub PE router to be able to distinguish between packets going to and coming from the spoke PE routers, you must configure it with two routing instances:

- One routing instance (in this example, **Spokes-to-Hub-CE**) is associated with the interface that carries packets from the hub PE router to the hub CE router (in this example, interface **ge-0/0/0.0**). Its VRF table contains the routes being announced by the spoke PE routers and the hub PE router to the hub CE router.
- The second routing instance (in this example, **Hub-CE-to-Spokes**) is associated with the interface that carries packets from the hub CE router to the hub PE router (in this example, interface **ge-0/0/1.0**). Its VRF table contains the routes being announced from the hub CE router to the hub-and-spoke PE routers.

On each spoke router, you must configure one routing instance.

You must define the following in the routing instance:

- Route distinguisher, which is used to distinguish the addresses in one VPN from those in another VPN.
- Instance type of **vrf**, which creates the VRF table on the PE router.
- Interfaces that are part of the VPN and that connect the PE routers to their CE routers.
- VRF import and export policies. Both import policies must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails. (The exception to this is if the import policy contains only a **then reject** statement.) In the VRF export policy, spoke PE routers attach the spoke target community.
- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

For a hub-and-spoke topology, you must configure different policies in each routing instance on the hub CE router. For the routing instance associated with the interface that carries packets from the hub PE router to the hub CE router (in this example, **Spokes-to-Hub-CE**), the import policy must accept all routes received on the IBGP session between the hub-and-spoke PE routers, and the export policy must reject all routes received from the hub CE router. For the routing instance associated with the interface that carries packets from the hub CE router to the hub PE router (in this example,

**Hub-CE-to-Spokes**), the import policy must reject all routes received from the spoke PE routers, and the export policy must export to all the spoke routers.

On hub PE Router D, configure the following routing instances. Router D uses OSPF to distribute routes to and from the hub CE router.

```
[edit]
routing-instance {
  Spokes-to-Hub-CE {
    instance-type vrf;
    interface ge-0/0/0.0;
    route-distinguisher 10.255.1.174:65535;
    vrf-import spoke;
    vrf-export null;
    protocols {
      ospf {
        export redistribute-vpn;
        domain-vpn-tag 0;
        area 0.0.0.0 {
          interface ge-0/0/0;
        }
      }
    }
  }
  Hub-CE-to-Spokes {
    instance-type vrf;
    interface ge-0/0/1.0;
    route-distinguisher 10.255.1.174:65535;
    vrf-import null;
    vrf-export hub;
    protocols {
      ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
          interface ge-0/0/1.0;
        }
      }
    }
  }
}
```

On spoke PE Router E, configure the following routing instances. Router E uses OSPF to distribute routes to and from spoke CE Router CE1.

```
[edit]
routing-instance {
  Spoke-E-to-Hub {
    instance-type vrf;
    interface fe-0/1/0.0;
    route-distinguisher 10.255.14.80:65535;
    vrf-import hub;
    vrf-export spoke;
    protocols {
      ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
```

```

        interface fe-0/1/0.0;
      }
    }
  }
}

```

On spoke PE Router F, configure the following routing instances. Router F uses OSPF to distribute routes to and from spoke CE Router CE2.

```

[edit]
routing-instance {
  Spoke-F-to-Hub {
    instance-type vrf;
    interface fe-1/0/1.0;
    route-distinguisher 10.255.14.182:65535;
    vrf-import hub;
    vrf-export spoke;
    protocols {
      ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
          interface fe-1/0/1.0;
        }
      }
    }
  }
}
}

```

## Configuring VPN Policy on the PE Routers

You must configure VPN import and export policies on each of the hub-and-spoke PE routers so that they install the appropriate routes in the VRF tables, which they use to forward packets within each VPN.

On the spoke routers, you define policies to exchange routes with the hub router.

On the hub router, you define policies to accept routes from the spoke PE routers and distribute them to the hub CE router, and vice versa. The hub PE router has two VRF tables:

- Spoke-to-hub VRF table—Handles routes received from spoke routers and announces these routes to the hub CE router. For this VRF table, the import policy must check that the spoke target name is present and that the route was received from the IBGP session between the hub PE and the spoke PE routers. This VRF table must not export any routes, so its export policy should reject everything.
- Hub-to-spoke VRF table—Handles routes received from the hub CE router and announces them to the spoke routers. For this VRF table, the export policy must add the hub target community. This VRF table must not import any routes, so its import policy should reject everything.

In the VPN policy, you also configure the VPN target communities.

On hub PE Router D, configure the following policies to apply to the VRF tables:

- **spoke**—Accepts routes received from the IBGP session between it and the spoke PE routers that contain the community target **spoke**, and rejects all other routes.
- **hub**—Adds the community target hub to all routes received from OSPF (that is, from the session between it and the hub CE router). It rejects all other routes.
- **null**—Rejects all routes.
- **redistribute-vpn**—Redistributes OSPF routes to neighbors within the routing instance.

```
[edit]
policy-options {
  policy-statement spoke {
    term a {
      from {
        protocol bgp;
        community spoke;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement hub {
    term a {
      from protocol ospf;
      then {
        community add hub;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  policy-statement null {
    then reject;
  }
  policy-statement redistribute-vpn {
    term a {
      from protocol bgp;
      then accept;
    }
    term b {
      then reject;
    }
  }
  community hub members target:65535:1;
  community spoke members target:65535:2;
}
```

To apply the VRF policies on Router D, include the **vrf-export** and **vrf-import** statements when you configure the routing instances:

```
[edit]
routing-instance {
```

```

Spokes-to-Hub-CE {
  vrf-import spoke;
  vrf-export null;
}
Hub-CE-to-Spokes {
  vrf-import null;
  vrf-export hub;
}
}

```

On spoke PE Router E and Router F, configure the following policies to apply to the VRF tables:

- **hub**—Accepts routes received from the IBGP session between it and the hub PE routers that contain the community target **hub**, and rejects all other routes.
- **spoke**—Adds the community target spoke to all routes received from OSPF (that is, from the session between it and the hub CE router) rejects all other routes.
- **redistribute-vpn**—Redistributes OSPF routes to neighbors within the routing instance.

On spoke PE Router E and Router F, configure the following VPN import and export policies:

```

[edit]
policy-options {
  policy-statement hub {
    term a {
      from {
        protocol bgp;
        community hub;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement spoke {
    term a {
      from protocol ospf;
      then {
        community add spoke;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  policy-statement redistribute-vpn {
    term a {
      from protocol bgp;
      then accept;
    }
    term b {

```

```

        then reject;
    }
}
community hub members target:65535:1;
community spoke members target 65535:2;
}

```

To apply the VRF policies on the spoke routers, include the **vrf-export** and **vrf-import** statements when you configure the routing instances:

```

[edit]
routing-instance {
  Spoke-E-to-Hub {
    vrf-import hub;
    vrf-export spoke;
  }
}
[edit]
routing-instance {
  Spoke-F-to-Hub {
    vrf-import hub;
    vrf-export spoke;
  }
}

```

## Hub-and-Spoke VPN Configuration Summarized by Router

### Router D (Hub PE Router)

Routing Instance for Distributing Spoke Routes to Hub CE	<pre> Spokes-to-Hub-CE {   instance-type vrf;   interface fe-0/0/0.0;   route-distinguisher 10.255.1.174:65535;   vrf-import spoke;   vrf-export null; } </pre>
Instance Routing Protocol	<pre> protocols {   ospf {     domain-vpn-tag 0;     export redistribute-vpn;     area 0.0.0.0 {       interface ge-0/0/0.0;     }   } } </pre>
Routing Instance for Distributing Hub CE Routes to Spokes	<pre> Hub-CE-to-Spokes {   instance-type vrf;   interface ge-0/0/1.0;   route-distinguisher 10.255.1.174:65535;   vrf-import null;   vrf-export hub; } </pre>

Routing Instance Routing Protocols	<pre> protocols {   ospf {     export redistribute-vpn;     area 0.0.0.0 {       interface ge-0/0/1.0;     }   } } </pre>
Routing Options (Master Instance)	<pre> routing-options {   autonomous-system 1 loops 1; } </pre>
Protocols (Master Instance)	<pre> protocols { } </pre>
Enable LDP	<pre> ldp {   interface so-1/0/0.0;   interface t3-1/1/0.0; } </pre>
Configure IBGP	<pre> bgp {   group Hub-to-Spokes {     type internal;     local-address 10.255.14.174;     family inet-vpn {       unicast;     }     neighbor 10.255.14.180;     neighbor 10.255.14.182;   } } </pre>
Configure VPN Policy	<pre> policy-options {   policy-statement spoke {     term a {       from {         protocol bgp;         community spoke;       }       then accept;     }     term b {       then reject;     }   }   policy-statement hub {     term a {       from protocol ospf;       then {         community add hub;         accept;       }     }     term b { </pre>

```

        then reject;
    }
}
policy-statement null {
    then reject;
}
policy-statement redistribute-vpn {
    term a {
        from protocol bgp;
        then accept;
    }
    term b {
        then reject;
    }
}
community hub members target:65535:1;
community spoke members target:65535:2;
}

```

### Router E (Spoke PE Router)

<b>Routing Instance</b>	<pre> routing-instance {     Spoke-E-to-Hub {         instance-type vrf;         interface fe-0/1/0.0;         route-distinguisher 10.255.14.80:65535;         vrf-import hub;         vrf-export spoke;     } } </pre>
<b>Instance Routing Protocol</b>	<pre> protocols {     ospf {         export redistribute-vpn;         area 0.0.0.0 {             interface fe-0/1/0.0;         }     } } </pre>
<b>Routing Options (Master Instance)</b>	<pre> routing-options {     autonomous-system 1 loops 1; } </pre>
<b>Protocols (Master Instance)</b>	<pre> protocols { } </pre>
<b>Enable LDP</b>	<pre> ldp {     interface fe-0/1/2.0; } </pre>
<b>Configure IBGP</b>	<pre> bgp {     group Spoke-E-to-Hub {         type internal;         local-address 10.255.14.180;     } } </pre>

```

neighbor 10.255.14.174 {
    family inet-vpn {
        unicast;
    }
}

```

**Configure VPN Policy**

```

policy-options {
    policy-statement hub {
        term a {
            from {
                protocol bgp;
                community hub;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement spoke {
        term a {
            from protocol ospf;
            then {
                community add spoke;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement redistribute-vpn {
        term a {
            from protocol bgp;
            then accept;
        }
        term b {
            then reject;
        }
    }
    community hub members target:65535:1;
    community spoke members target:65535:2;
}

```

**Router F (Spoke PE Router)****Routing Instance**

```

routing-instance {
    Spoke-F-to-Hub {
        instance-type vrf;
        interface fe-1/0/1.0;
        route-distinguisher 10.255.14.182:65535;
        vrf-import hub;
        vrf-export spoke;
    }
}

```

	<pre>         }       } </pre>
<b>Instance Routing Protocol</b>	<pre> protocols {   ospf {     export redistribute-vpn;     area 0.0.0.0 {       interface fe-1/0/1.0;     }   } } </pre>
<b>Routing Options (Master Instance)</b>	<pre> routing-options {   autonomous-system 1 loops 1; } </pre>
<b>Protocols (Master Instance)</b>	<pre> protocols { } </pre>
<b>Enable LDP</b>	<pre> ldp {   interface fe-1/0/0.0; } </pre>
<b>Configure IBGP</b>	<pre> bgp {   group Spoke-F-to-Hub {     type internal;     local-address 10.255.14.182;     neighbor 10.255.14.174 {       family inet-vpn {         unicast;       }     }   } } </pre>
<b>Configure VPN Policy</b>	<pre> policy-options {   policy-statement hub {     term a {       from {         protocol bgp;         community hub;       }       then accept;     }     term b {       then reject;     }   }   policy-statement spoke {     term a {       from protocol ospf;       then {         community add spoke;         accept;       }     }   } } </pre>

```

    }
    term b {
        then reject;
    }
}
policy-statement redistribute-vpn {
    term a {
        from {
            protocol bgp;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
community hub members target:65535:1;
community spoke members target:65535:2;
}

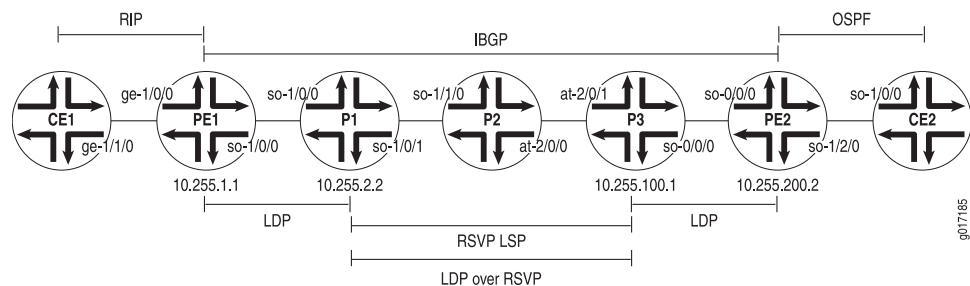
```

## Configuring an LDP-over-RSVP VPN Topology

This example shows how to set up a VPN topology in which LDP packets are tunneled over an RSVP LSP. This configuration consists of the following components (see [Figure 24 on page 159](#)):

- One VPN (VPN-A)
- Two PE routers
- LDP as the signaling protocol between the PE routers and their adjacent P routers
- An RSVP LSP between two of the P routers over which LDP is tunneled

**Figure 24: Example of an LDP-over-RSVP VPN Topology**



The following steps describe how this topology is established and how packets are sent from CE Router CE2 to CE Router CE1:

1. The P routers P1 and P3 establish RSVP LSPs between each other and install their loopback addresses in their inet.3 routing tables.
2. PE Router PE1 establishes an LDP session with Router P1 over interface **so-1/0/0.0**.
3. Router P1 establishes an LDP session with Router P3's loopback address, which is reachable using the RSVP LSP.

4. Router P1 sends its label bindings, which include a label to reach Router PE1, to Router P3. These label bindings allow Router P3 to direct LDP packets to Router PE1.
5. Router P3 establishes an LDP session with Router PE2 over interface **so0-0/0/0.0** and establishes an LDP session with Router P1's loopback address.
6. Router P3 sends its label bindings, which include a label to reach Router PE2, to Router P1. These label bindings allow Router P1 to direct LDP packets to Router PE2's loopback address.
7. Routers PE1 and PE2 establish IBGP sessions with each other.
8. When Router PE1 announces to Router PE2 routes that it learned from Router CE1, it includes its VPN label. (The PE router creates the VPN label and binds it to the interface between the PE and CE routers.) Similarly, when Router PE2 announces routes that it learned from Router CE2, it sends its VPN label to Router PE1.

When Router PE2 wants to forward a packet to Router CE1, it pushes two labels onto the packet's label stack: first the VPN label that is bound to the interface between Router PE1 and Router CE1, then the LDP label used to reach Router PE1. Then it forwards the packets to Router P3 over interface **so-0/0/1.0**.

1. When Router P3 receives the packets from Router PE2, it swaps the LDP label that is on top of the stack (according to its LDP database) and also pushes an RSVP label onto the top of the stack so that the packet can now be switched by the RSVP LSP. At this point, there are three labels on the stack: the inner (bottom) label is the VPN label, the middle is the LDP label, and the outer (top) is the RSVP label.
2. Router P2 receives the packet and switches it to Router P1 by swapping the RSVP label. In this topology, because Router P2 is the penultimate-hop router in the LSP, it pops the RSVP label and forwards the packet over interface **so-1/1/0.0** to Router P1. At this point, there are two labels on the stack: The inner label is the VPN label, and the outer one is the LDP label.
3. When Router P1 receives the packet, it pops the outer label (the LDP label) and forwards the packet to Router PE1 using interface **so-1/0/0.0**. In this topology, Router PE1 is the egress LDP router, so Router P1 pops the LDP label instead of swapping it with another label. At this point, there is only one label on the stack, the VPN label.
4. When Router PE1 receives the packet, it pops the VPN label and forwards the packet as an IPv4 packet to Router CE1 over interface **ge-1/1/0.0**.

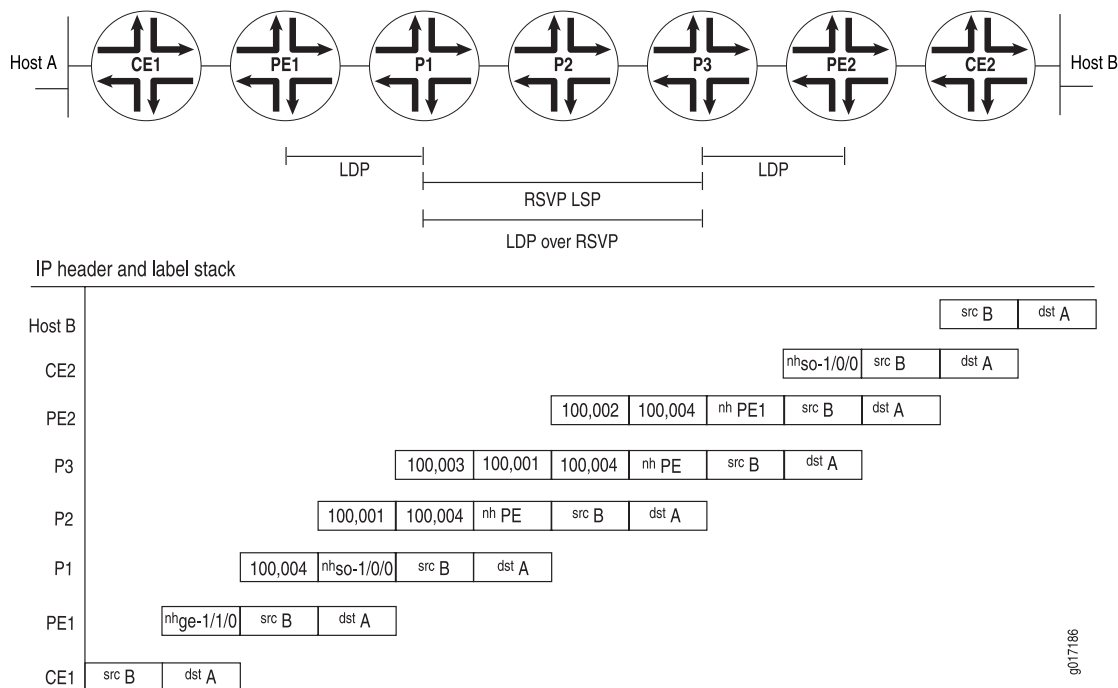
A similar set of operations occurs for packets sent from Router CE1 that are destined for Router CE2.

The following list explains how, for packets being sent from Router CE2 to Router CE1, the LDP, RSVP, and VPN labels are announced by the various routers. These steps include examples of label values (illustrated in [Figure 25 on page 161](#)).

- LDP labels
  - Router PE1 announces LDP label 3 for itself to Router P1.
  - Router P1 announces LDP label 100,001 for Router PE1 to Router P3.

- Router P3 announces LDP label 100,002 for Router PE1 to Router PE2.
- RSVP labels
  - Router P1 announces RSVP label 3 to Router P2.
  - Router P2 announces RSVP label 100,003 to Router P3.
- VPN label
  - Router PE1 announces VPN label 100,004 to Router PE2 for the route from Router CE1 to Router CE2.

Figure 25: Label Pushing and Popping



For a packet sent from Host B in [Figure 25 on page 161](#) to Host A, the packet headers and labels change as the packet travels to its destination:

1. The packet that originates from Host B has a source address of B and a destination address of A in its header.
2. Router CE2 adds to the packet a next hop of interface **so-1/0/0**.
3. Router PE2 swaps out the next hop of interface **so-1/0/0** and replaces it with a next hop of PE1. It also adds two labels for reaching Router PE1, first the VPN label (100,004), then the LDP label (100,002). The VPN label is thus the inner (bottom) label on the stack, and the LDP label is the outer label.
4. Router P3 swaps out the LDP label added by Router PE2 (100,002) and replaces it with its LDP label for reaching Router PE1 (100,001). It also adds the RSVP label for reaching Router P2 (100,003).

5. Router P2 removes the RSVP label (100,003) because it is the penultimate hop in the MPLS LSP.
6. Router P1 removes the LDP label (100,001) because it is the penultimate LDP router. It also swaps out the next hop of PE1 and replaces it with the next-hop interface, **so-1/0/0**.
7. Router PE1 removes the VPN label (100,004). It also swaps out the next-hop interface of **so-1/0/0** and replaces it with its next-hop interface, **ge-1/1/0**.
8. Router CE1 removes the next-hop interface of **ge-1/1/0**, and the packet header now contains just a source address of B and a destination address of A.

The final section in this example consolidates the statements needed to configure VPN functionality on each of the service P routers shown in [Figure 24 on page 159](#).



**NOTE:** In this example, a private AS number is used for the route distinguisher and the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

The following sections explain how to configure the VPN functionality on the PE and P routers. The CE routers do not have any information about the VPN, so you configure them normally.

- [Enabling an IGP on the PE and P Routers on page 162](#)
- [Enabling LDP on the PE and P Routers on page 162](#)
- [Enabling RSVP and MPLS on the P Router on page 164](#)
- [Configuring the MPLS LSP Tunnel Between the P Routers on page 164](#)
- [Configuring IBGP on the PE Routers on page 165](#)
- [Configuring Routing Instances for VPNs on the PE Routers on page 166](#)
- [Configuring VPN Policy on the PE Routers on page 167](#)
- [LDP-over-RSVP VPN Configuration Summarized by Router on page 169](#)

## Enabling an IGP on the PE and P Routers

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

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

## Enabling LDP on the PE and P Routers

In this configuration example, the LDP is the signaling protocol between the PE routers. For the VPN to function, you must configure LDP on the two PE routers and on the P routers that are connected to the PE routers. You need to configure LDP only on the

interfaces in the core of the service provider's network; that is, between the PE and P routers and between the P routers. You do not need to configure LDP on the interface between the PE and CE routers.

In this configuration example, you configure LDP on the P routers' loopback interfaces because these are the interfaces on which the MPLS LSP is configured.

On the PE routers, you must also configure **family inet** when you configure the logical interface.

On Router PE1, configure LDP:

```
[edit protocols]
ldp {
  interface so-1/0/0.0;
}
[edit interfaces]
so-1/0/0 {
  unit 0 {
    family mpls;
  }
}
```

On Router PE2, configure LDP:

```
[edit protocols]
ldp {
  interface so-0/0/0.0;
}
[edit interfaces]
so-0/0/1 {
  unit 0 {
    family mpls;
  }
}
```

On Router P1, configure LDP:

```
[edit protocols]
ldp {
  interface so-1/0/0.0;
  interface lo0;
}
```

On Router P3, configure LDP:

```
[edit protocols]
ldp {
  interface lo0;
  interface so-0/0/0.0;
}
```

On Router P2, although you do not need to configure LDP, you can optionally configure it to provide a fallback LDP path in case the RSVP LSP becomes nonoperational:

```
[edit protocols]
ldp {
```

```
interface so-1/1/0.0;  
interface at-2/0/0.0;  
}
```

## Enabling RSVP and MPLS on the P Router

On the P Router P2 you must configure RSVP and MPLS because this router exists on the MPLS LSP path between the P Routers P1 and P3:

```
[edit]  
protocols {  
  rsvp {  
    interface so-1/1/0.0;  
    interface at-2/0/0.0;  
  }  
  mpls {  
    interface so-1/1/0.0;  
    interface at-2/0/0.0;  
  }  
}
```

## Configuring the MPLS LSP Tunnel Between the P Routers

In this configuration example, LDP is tunneled over an RSVP LSP. Therefore, in addition to configuring RSVP, you must enable traffic engineering support in an IGP, and you must create an MPLS LSP to tunnel the LDP traffic.

On Router P1, enable RSVP and configure one end of the MPLS LSP tunnel. In this example, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces to the LSP and to Router PE1. In the **to** statement, you specify the loopback address of Router P3.

```
[edit]  
protocols {  
  rsvp {  
    interface so-1/0/1.0;  
  }  
  mpls {  
    label-switched-path P1-to-P3 {  
      to 10.255.100.1;  
      ldp-tunneling;  
    }  
    interface so-1/0/0.0;  
    interface so-1/0/1.0;  
  }  
  ospf {  
    traffic-engineering;  
    area 0.0.0.0 {  
      interface so-1/0/0.0;  
      interface so-1/0/1.0;  
    }  
  }  
}
```

On Router P3, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces

to the LSP and to Router PE2. In the **to** statement, you specify the loopback address of Router P1.

```
[edit]
protocols {
  rsvp {
    interface at-2/0/1.0;
  }
  mpls {
    label-switched-path P3-to-P1 {
      to 10.255.2.2;
      ldp-tunneling;
    }
    interface at-2/0/1.0;
    interface so-0/0/0.0;
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface at-2/0/1.0;
      interface so-0/0/0.0;
    }
  }
}
```

## Configuring IBGP on the PE Routers

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

- VPN family—To indicate that the IBGP session is for the VPN, include the **family inet-vpn** statement.
- Loopback address—Include the **local-address** statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the **lo0** interface at the **[edit interfaces]** hierarchy level. The example does not include this part of the router's configuration.
- Neighbor address—Include the **neighbor** statement, specifying the IP address of the neighboring PE router, which is its loopback (**lo0**) address.

On Router PE1, configure IBGP:

```
[edit]
protocols {
  bgp {
    group PE1-to-PE2 {
      type internal;
      local-address 10.255.1.1;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.200.2;
    }
  }
}
```

On Router PE2, configure IBGP:

```
[edit]
protocols {
  bgp {
    group PE2-to-PE1 {
      type internal;
      local-address 10.255.200.2;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.1.1;
    }
  }
}
```

## Configuring Routing Instances for VPNs on the PE Routers

Both PE routers service VPN-A, so you must configure one routing instance on each router for the VPN in which you define the following:

- Route distinguisher, which must be unique for each routing instance on the PE router. It is used to distinguish the addresses in one VPN from those in another VPN.
- Instance type of **vrf**, which creates the VRF table on the PE router.
- Interfaces connected to the CE routers.
- VRF import and export policies, which must be the same on each PE router that services the same VPN. Unless the import policy contains only a **then reject** statement, it must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails.



**NOTE:** In this example, a private AS number is used for the route distinguisher. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

On Router PE1, configure the following routing instance for VPN-A. In this example, Router PE1 uses RIP to distribute routes to and from the CE router to which it is connected.

```
[edit]
routing-instance {
  VPN-A {
    instance-type vrf;
    interface ge-1/0/0.0;
    route-distinguisher 65535:0;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    protocols {
      rip {
```

```

        group PE1-to-CE1 {
            neighbor ge-1/0/0.0;
        }
    }
}

```

On Router PE2, configure the following routing instance for VPN-A. In this example, Router PE2 uses OSPF to distribute routes to and from the CE router to which it is connected.

```

[edit]
routing-instance {
  VPN-A {
    instance-type vrf;
    interface so-1/2/0.0;
    route-distinguisher 65535:1;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    protocols {
      ospf {
        area 0.0.0.0 {
          interface so-1/2/0.0;
        }
      }
    }
  }
}

```

## Configuring VPN Policy on the PE Routers

You must configure VPN import and export policies on each of the PE routers so that they install the appropriate routes in their VRF tables, which they use to forward packets within a VPN. For VPN-A, the VRF table is VPN-A.inet.0.

In the VPN policy, you also configure VPN target communities.



**NOTE:** In this example, a private AS number is used for the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

On Router PE1, configure the following VPN import and export policies:



**NOTE:** The policy qualifiers shown in this example are only those needed for the VPN to function. You can configure additional qualifiers, as needed, to any policies that you configure.

```

[edit]
policy-options {
  policy-statement VPN-A-import {

```

```
term a {
  from {
    protocol bgp;
    community VPN-A;
  }
  then accept;
}
term b {
  then reject;
}
}
policy-statement VPN-A-export {
  term a {
    from protocol rip;
    then {
      community add VPN-A;
      accept;
    }
  }
  term b {
    then reject;
  }
}
community VPN-A members target:65535:00;
}
```

On Router PE2, configure the following VPN import and export policies:

```
[edit]
policy-options {
  policy-statement VPN-A-import {
    term a {
      from {
        protocol bgp;
        community VPN-A;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement VPN-A-export {
    term a {
      from protocol ospf;
      then {
        community add VPN-A;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  community VPN-A members target:65535:00;
}
```

To apply the VPN policies on the routers, include the **vrf-export** and **vrf-import** statements when you configure the routing instance on the PE routers. The VRF import and export policies handle the route distribution across the IBGP session running between the PE routers.

## LDP-over-RSVP VPN Configuration Summarized by Router

Router PE1	
Routing Instance for VPN-A	<pre> routing-instance {   VPN-A {     instance-type vrf;     interface ge-1/0/0.0;     route-distinguisher 65535:0;     vrf-import VPN-A-import;     vrf-export VPN-A-export;   } } </pre>
Instance Routing Protocol	<pre> protocols {   rip {     group PE1-to-CE1 {       neighbor ge-1/0/0.0;     }   } } </pre>
Interfaces	<pre> interfaces {   so-1/0/0 {     unit 0 {       family mpls;     }   }   ge-1/0/0 {     unit 0;   } } </pre>
Master Protocol Instance	<pre> protocols { } </pre>
Enable LDP	<pre> ldp {   interface so-1/0/0.0; } </pre>
Enable MPLS	<pre> mpls {   interface so-1/0/0.0;   interface ge-1/0/0.0; } </pre>
Configure IBGP	<pre> bgp {   group PE1-to-PE2 {     type internal;     local-address 10.255.1.1;   } } </pre>

```

        family inet-vpn {
            unicast;
        }
        neighbor 10.255.100.1;
    }
}

Configure VPN Policy
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol rip;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:00;
}

```

### Router P1

<b>Master Protocol Instance</b>	<pre> protocols { } </pre>
<b>Enable RSVP</b>	<pre> rsvp {     interface so-1/0/1.0; } </pre>
<b>Enable LDP</b>	<pre> ldp {     interface so-1/0/0.0;     interface lo0.0; } </pre>
<b>Enable MPLS</b>	<pre> mpls {     label-switched-path P1-to-P3 {         to 10.255.100.1;         ldp-tunneling;     }     interface so-1/0/0.0; } </pre>

	<pre> interface so-1/0/1.0; } </pre>
Configure OSPF for Traffic Engineering Support	<pre> ospf {   traffic-engineering;   area 0.0.0.0 {     interface so-1/0/0.0;     interface so-1/0/1.0;   } } </pre>

#### Router P2

---

Master Protocol Instance	<pre> protocols { } </pre>
Enable RSVP	<pre> rsvp {   interface so-1/1/0.0;   interface at-2/0/0.0; } </pre>
Enable MPLS	<pre> mpls {   interface so-1/1/0.0;   interface at-2/0/0.0; } </pre>

#### Router P3

---

Master Protocol Instance	<pre> protocols { } </pre>
Enable RSVP	<pre> rsvp {   interface at-2/0/1.0; } </pre>
Enable LDP	<pre> ldp {   interface so-0/0/0.0;   interface lo0.0; } </pre>
Enable MPLS	<pre> mpls {   label-switched-path P3-to-P1 {     to 10.255.2.2;     ldp-tunneling;   }   interface at-2/0/1.0;   interface so-0/0/0.0; } </pre>
Configure OSPF for Traffic Engineering Support	<pre> ospf {   traffic-engineering;   area 0.0.0.0 {     interface at-2/0/1.0;     interface at-2/0/1.0;   } } </pre>

```

    }
  }

```

## Router PE2

Routing Instance for VPN-A	<pre> routing-instance {   VPN-A {     instance-type vrf;     interface so-1/2/0.0;     route-distinguisher 65535:1;     vrf-import VPN-A-import;     vrf-export VPN-A-export;   } } </pre>
Instance Routing Protocol	<pre> protocols {   ospf {     area 0.0.0.0 {       interface so-1/2/0.0;     }   } } </pre>
Interfaces	<pre> interfaces {   so-0/0/0 {     unit 0 {       family mpls;     }   }   so-1/2/0 {     unit 0;   } } </pre>
Master Protocol Instance	<pre> protocols { } </pre>
Enable LDP	<pre> ldp {   interface so-0/0/0.0; } </pre>
Enable MPLS	<pre> mpls {   interface so-0/0/0.0;   interface so-1/2/0.0; } </pre>
Configure IBGP	<pre> bgp {   group PE2-to-PE1 {     type internal;     local-address 10.255.200.2;     family inet-vpn {       unicast;     }   }   neighbor 10.255.1.1; } </pre>

```

    }
  }
}

Configure VPN Policy
policy-options {
  policy-statement VPN-A-import {
    term a {
      from {
        protocol bgp;
        community VPN-A;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement VPN-A-export {
    term a {
      from protocol ospf;
      then {
        community add VPN-A;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  community VPN-A members target:65535:01;
}

```

## Configuring an Application-Based Layer 3 VPN Topology

This example illustrates an application-based mechanism for forwarding traffic into a Layer 3 VPN. Typically, one or more interfaces are associated with, or bound to, a VPN by including them in the configuration of the VPN routing instance. By binding the interface to the VPN, the VPN's VRF table is used to make forwarding decisions for any incoming traffic on that interface. Binding the interface also includes the interface local routes in the VRF table, which provides next-hop resolution for VRF routes.

In this example, a firewall filter is used to define which incoming traffic on an interface is forwarded by means of the standard routing table, inet.0, and which incoming traffic is forwarded by means of the VRF table. You can expand this example such that incoming traffic on an interface can be redirected to one or more VPNs. For example, you can define a configuration to support a VPN that forwards traffic based on source address, that forwards Hypertext Transfer Protocol (HTTP) traffic, or that forwards only streaming media.

For this configuration to work, the following conditions must be true:

- The interfaces that use filter-based forwarding must not be bound to the VPN.
- Static routing must be used as the means of routing.

- You must define an interface routing table group that is shared among inet.0 and the VRF tables to provide local routes to the VRF table.

This example consists of two client hosts (Client D and Client E) that are in two different VPNs and that want to send traffic both within the VPN and to the Internet. The paths are defined as follows:

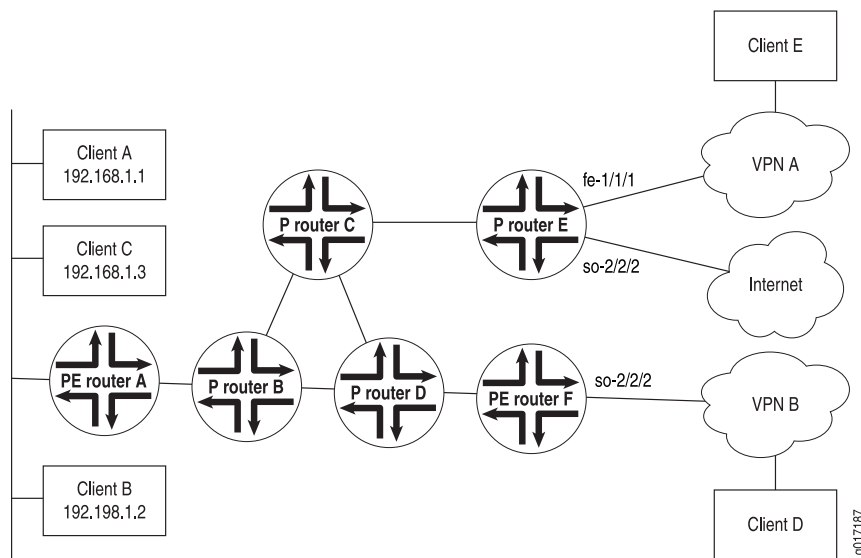
- Client A sends traffic to Client E over VPN A with a return path that also uses VPN A (using the VPN's VRF table).
- Client B sends traffic to Client D over VPN B with a return path that uses standard destination-based routing (using the inet.0 routing table).
- Clients B and C send traffic to the Internet using standard routing (using the inet.0 routing table), with a return path that also uses standard routing.

This example illustrates that there are a large variety of options in configuring an application-based Layer 3 VPN topology. This flexibility has application in many network implementations that require specific traffic to be forwarded in a constrained routing environment.

This configuration example shows only the portions of the configuration for the filter-based forwarding, routing instances, and policy. It does not illustrate how to configure a Layer 3 VPN.

Figure 26 on page 174 illustrates the network topology used in this example.

**Figure 26: Application-Based Layer 3 VPN Example Configuration**



- [Configuration on Router A on page 175](#)
- [Configuration on Router E on page 176](#)
- [Configuration on Router F on page 177](#)

## Configuration on Router A

On Router A, you configure the interface to Clients A, B, and C. The configuration evaluates incoming traffic to determine whether it is to be forwarded by means of VPN or standard destination-based routing.

First, you apply an inbound filter and configure the interface:

```
[edit]
interfaces {
  fe-1/1/0 {
    unit 0 {
      family inet {
        filter {
          input fbf-vrf;
        }
        address 192.168.1.1/24;
      }
    }
  }
}
```

Because the interfaces that use filter-based forwarding must not be bound to a VPN, you must configure an alternate method to provide next-hop routes to the VRF table. You do this by defining an interface routing table group and sharing this group among all the routing tables:

```
[edit]
routing-options {
  interface-routes {
    rib-group inet if-rib;
  }
  rib-groups {
    if-rib {
      import-rib [ inet.0 vpn-A.inet.0 vpn-B.inet.0 ];
    }
  }
}
```

You apply the following filter to incoming traffic on interface **fe-1/1/0.0**. The first term matches traffic from Client A and forwards it to the routing instance for VPN A. The second term matches traffic from Client B that is destined for Client D and forwards it to the routing instance for VPN B. The third term matches all other traffic, which is forwarded normally by means of destination-based forwarding according to the routes in inet.0.

```
[edit firewall family family-name]
filter fbf-vrf {
  term vpnA {
    from {
      source-address {
        192.168.1.1/32;
      }
    }
    then {
```

```

        routing-instance vpn-A;
    }
}
term vpnB {
    from {
        source-address {
            192.168.1.2/32;
        }
        destination-address {
            192.168.3.0/24;
        }
    }
    then routing-instance vpn-B;
}
}
term internet {
    then accept;
}
}

```

You then configure the routing instances for VPN A and VPN B. Notice that these statements include all the required statements to define a Layer 3 VPN except for the **interface** statement.

```

[edit]
routing-instances {
    vpn-A {
        instance-type vrf;
        route-distinguisher 172.21.10.63:100;
        vrf-import vpn-A-import;
        vrf-export vpn-A-export;
    }
    vpn-B {
        instance-type vrf;
        route-distinguisher 172.21.10.63:200;
        vrf-import vpn-B-import;
        vrf-export vpn-B-export;
    }
}
}

```

## Configuration on Router E

On Router E, configure a default route to reach the Internet. You should inject this route into the local IBGP mesh to provide an exit point from the network.

```

[edit]
routing-options {
    static {
        route 0.0.0.0/0 next-hop so-2/2/2.0 discard
    }
}
}

```

Configure the interface to Client E so that all incoming traffic on interface **fe-1/1/1.0** that matches the VPN policy is forwarded over VPN A:

```

[edit]
routing-instances {

```

```

vpn-A {
  interface fe-1/1/1.0
  instance-type vrf;
  route-distinguisher 172.21.10.62:100;
  vrf-import vpn-A-import;
  vrf-export vpn-A-export;
  routing-options {
    static {
      route 192.168.2.0/24 next-hop fe-1/1/1.0;
    }
  }
}

```

### Configuration on Router F

Again, because the interfaces that use filter-based forwarding must not be bound to a VPN, you configure an alternate method to provide next-hop routes to the VRF table by defining an interface routing table group and sharing this group among all the routing tables. To provide a route back to the clients for normal inet.0 routing, you define a static route to include in inet.0 and redistribute the static route into BGP:

```

[edit]
routing-options {
  interface-routes {
    rib-group inet if-rib;
  }
  rib-groups {
    if-rib {
      import-rib [ inet.0 vpn-B.inet.0 ];
    }
  }
}

```

To direct traffic from VPN B to Client D, you configure the routing instance for VPN B on Router F. All incoming traffic from Client D on interface **so-3/3/3.0** is forwarded normally by means of the destination address based on the routes in inet.0.

```

[edit]
routing-instances {
  vpn-B {
    instance-type vrf;
    route-distinguisher 172.21.10.64:200;
    vrf-import vpn-B-import;
    vrf-export vpn-B-export;
    routing-options {
      static {
        route 192.168.3.0/24 next-hop so-3/3/3.0;
      }
    }
  }
}

```

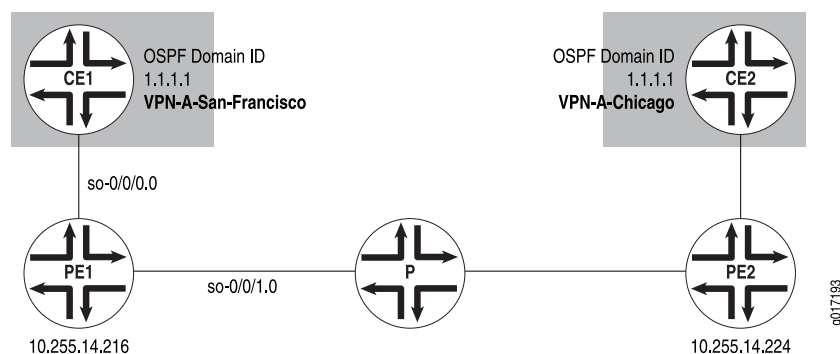
## Configuring an OSPF Domain ID for a Layer 3 VPN

This example illustrates how to configure an OSPF domain ID for a VPN by using OSPF as the routing protocol between the PE and CE routers. Routes from an OSPF domain need an OSPF domain ID when they are distributed in BGP as VPN-IPv4 routes in VPNs with multiple OSPF domains. In a VPN connecting multiple OSPF domains, the routes from one domain might overlap with the routes of another.

For more information about OSPF domain IDs and Layer 3 VPNs, see [“Configuring Routing Between PE and CE Routers in Layer 3 VPNs” on page 31](#).

[Figure 27 on page 178](#) shows this example's configuration topology. Only the configuration for Router PE1 is provided. The configuration for Router PE2 can be similar to the configuration for Router PE1. There are no special configuration requirements for the CE routers.

**Figure 27: Example of a Configuration Using an OSPF Domain ID**



For configuration information, see the following sections:

- [Configuring Interfaces on Router PE1 on page 178](#)
- [Configuring Routing Options on Router PE1 on page 179](#)
- [Configuring Protocols on Router PE1 on page 179](#)
- [Configuring Policy Options on Router PE1 on page 180](#)
- [Configuring the Routing Instance on Router PE1 on page 180](#)
- [Configuration Summary for Router PE1 on page 181](#)

### Configuring Interfaces on Router PE1

You need to configure two interfaces for Router PE1—the **so-0/0/0** interface for traffic to Router CE1 (San Francisco) and the **so-0/0/1** interface for traffic to a P router in the service provider's network.

Configure the interfaces for Router PE1:

```
[edit]
interfaces {
  so-0/0/0 {
    unit 0 {
```

```

        family inet {
            address 10.19.1.2/30;
        }
    }
}
so-0/0/1 {
    unit 0 {
        family inet {
            address 10.19.2.1/30;
        }
        family mpls;
    }
}
}

```

### Configuring Routing Options on Router PE1

At the **[edit routing-options]** hierarchy level, you need to configure the **router-id** and **autonomous-system** statements. The **router-id** statement identifies Router PE1.

Configure the routing options for Router PE1:

```

[edit]
routing-options {
    router-id 10.255.14.216;
    autonomous-system 69;
}

```

### Configuring Protocols on Router PE1

On Router PE1, you need to configure MPLS, BGP, OSPF, and LDP at the **[edit protocols]** hierarchy level:

```

[edit]
protocols {
    mpls {
        interface so-0/0/1.0;
    }
    bgp {
        group San-Francisco-Chicago {
            type internal;
            preference 10;
            local-address 10.255.14.216;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.14.224;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-0/0/1.0;
        }
    }
    ldp {

```

```
        interface so-0/0/1.0;
    }
}
```

## Configuring Policy Options on Router PE1

On Router PE1, you need to configure policies at the **[edit policy-options]** hierarchy level. These policies ensure that the CE routers in the Layer 3 VPN exchange routing information. In this example, Router CE1 in San Francisco exchanges routing information with Router CE2 in Chicago.

Configure the policy options on the PE1 router:

```
[edit]
policy-options {
  policy-statement vpn-import-VPN-A {
    term term1 {
      from {
        protocol bgp;
        community import-target-VPN-A;
      }
      then accept;
    }
    term term2 {
      then reject;
    }
  }
  policy-statement vpn-export-VPN-A {
    term term1 {
      from protocol ospf;
      then {
        community add export-target-VPN-A;
        accept;
      }
    }
    term term2 {
      then reject;
    }
  }
  community export-target-VPN-A members [target:10.255.14.216:11
  domain-id:1.1.1.1:0];
  community import-target-VPN-A members target:10.255.14.224:31;
}
```

## Configuring the Routing Instance on Router PE1

You need to configure a Layer 3 VPN routing instance on Router PE1. To indicate that the routing instance is for a Layer 3 VPN, add the **instance-type vrf** statement at the **[edit routing-instance *routing-instance-name*]** hierarchy level.

The **domain-id** statement is configured at the **[edit routing-instances routing-options protocols ospf]** hierarchy level. As shown in [Figure 27 on page 178](#), the routing instance on Router PE2 must share the same domain ID as the corresponding routing instance on Router PE1 so that routes from Router CE1 to Router CE2 and vice versa are distributed as Type 3 LSAs. If you configure different OSPF domain IDs in the routing instances for

Router PE1 and Router PE2, the routes from each CE router will be distributed as Type 5 LSAs.

Configure the routing instance on Router PE1:

```
[edit]
routing-instances {
  VPN-A-San-Francisco-Chicago {
    instance-type vrf;
    interface so-0/0/0.0;
    route-distinguisher 10.255.14.216:11;
    vrf-import vpn-import-VPN-A;
    vrf-export vpn-export-VPN-A;
    routing-options {
      router-id 10.255.14.216;
      autonomous-system 69;
    }
    protocols {
      ospf {
        domain-id 1.1.1.1;
        export vpn-import-VPN-A;
        area 0.0.0.0 {
          interface so-0/0/0.0;
        }
      }
    }
  }
}
```

### Configuration Summary for Router PE1

<b>Configure Interfaces</b>	<pre>interfaces {   so-0/0/0 {     unit 0 {       family inet {         address 10.19.1.2/30;       }     }   }   so-0/0/1 {     unit 0 {       family inet {         address 10.19.2.1/30;       }       family mpls;     }   } }</pre>
<b>Configure Routing Options</b>	<pre>routing-options {   router-id 10.255.14.216;   autonomous-system 69; }</pre>
<b>Configure Protocols</b>	<pre>protocols {   mpls {</pre>

```

    interface so-0/0/0.0;
  }
  bgp {
    group San-Francisco-Chicago {
      type internal;
      preference 10;
      local-address 10.255.14.216;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.14.224;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-0/0/1.0;
    }
  }
  ldp {
    interface so-0/0/1.0;
  }
}

```

**Configure VPN Policy**

```

policy-options {
  policy-statement vpn-import-VPN-A {
    term term1 {
      from {
        protocol bgp;
        community import-target-VPN-A;
      }
      then accept;
    }
    term term2 {
      then reject;
    }
  }
  policy-statement vpn-export-VPN-A {
    term term1 {
      from protocol ospf;
      then {
        community add export-target-VPN-A;
        accept;
      }
    }
    term term2 {
      then reject;
    }
  }
  community export-target-VPN-B members [ target:10.255.14.216:1domain-id:1.1.1.1:0 ];
  community import-target-VPN-B members target:10.255.14.224:31;
}

```

**Routing Instance for  
Layer 3 VPN**

```

routing-instances {
  VPN-A-San-Francisco-Chicago {

```

```

instance-type vrf;
interface so-0/0/0.0;
route-distinguisher 10.255.14.216:11;
vrf-import vpn-import-VPN-A;
vrf-export vpn-export-VPN-A;
routing-options {
  router-id 10.255.14.216;
  autonomous-system 69;
}
protocols {
  ospf {
    domain-id 1.1.1.1;
    export vpn-import-VPN-A;
    area 0.0.0.0 {
      interface so-0/0/0.0;
    }
  }
}
}
}

```

## Configuring Overlapping VPNs Using Routing Table Groups

In Layer 3 VPNs, a CE router is often a member of more than one VPN. This example illustrates how to configure PE routers that support CE routers that support multiple VPNs. Support for this type of configuration uses a Junos OS feature called routing table groups (sometimes also called routing information base [RIB] groups), which allows a route to be installed into several routing tables. A routing table group is a list of routing tables into which the protocol should install its routes.

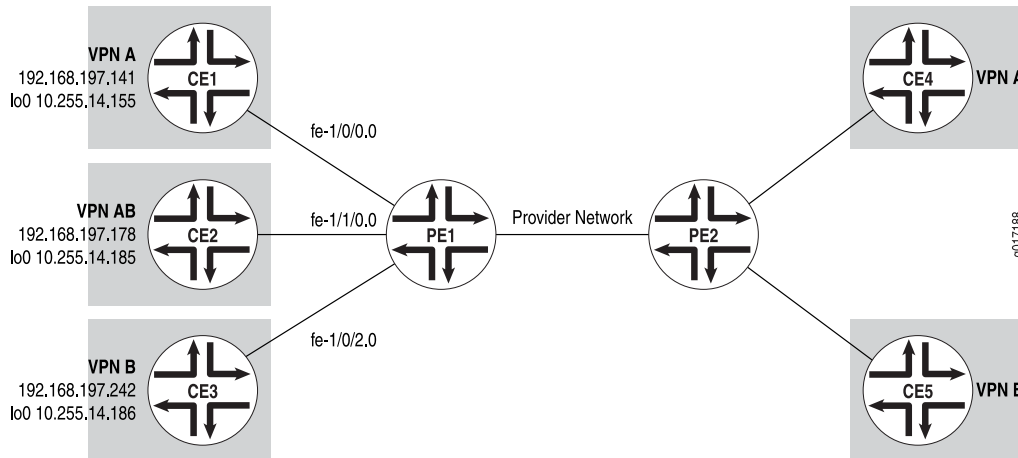
You define routing table groups at the **[edit routing-options]** hierarchy level for the default instance. You cannot configure routing table groups at the **[edit routing-instances routing-options]** hierarchy level; doing so results in a commit error.

After you define a routing table group, it can be used by multiple protocols. You can also apply routing table groups to static routing. The configuration examples in this section include both types of configurations.

[Figure 28 on page 184](#) illustrates the topology for the configuration example in this section. The configurations in this section illustrate local connectivity between CE routers connected to the same PE router. If Router PE1 were connected only to Router CE2 (VPN AB), there would be no need for any extra configuration. The configuration statements in the sections that follow enable VPN AB Router CE2 to communicate with VPN A Router CE1 and VPN B Router CE3, which are directly connected to Router PE1. VPN routes that originate from the remote PE routers (the PE2 router in this case) are placed in a global Layer 3 VPN routing table (bgp.l3vpn.inet.0), and routes with appropriate route targets are imported into the routing tables as dictated by the VRF import policy configuration. The goal is to be able to choose routes from individual VPN routing tables that are locally populated.

Router PE1 is where all the filtering and configuration modification takes place. Therefore only VPN configurations for PE1 are shown. The CE routers do not have any information about the VPN, so you can configure them normally.

Figure 28: Example of an Overlapping VPN Topology



The following sections explain several ways to configure overlapping VPNs.

The following sections illustrate different scenarios for configuring overlapping VPNs, depending on the routing protocol used between the PE and CE routers. For all of these examples, you need to configure routing table groups.

- [Configuring Routing Table Groups on page 184](#)
- [Configuring Static Routes Between the PE and CE Routers on page 185](#)
- [Configuring BGP Between the PE and CE Routers on page 190](#)
- [Configuring OSPF Between the PE and CE Routers on page 191](#)
- [Configuring Static, BGP, and OSPF Routes Between PE and CE Routers on page 193](#)

## Configuring Routing Table Groups

In this example, routing table groups are common in the four configuration scenarios. The routing table groups are used to install routes (including interface, static, OSPF, and BGP routes) into several routing tables for the default and other instances. In the routing table group definition, the first routing table is called the primary routing table. (Normally, the primary routing table is the table into which the route would be installed if you did not configure routing table groups. The other routing tables are called secondary routing tables.)

The routing table groups in this configuration install routes as follows:

- **vpna-vpnab** installs routes into routing tables VPN-A.inet.0 and VPN-AB.inet.0.
- **vpnb-vpnab** installs routes into routing tables VPN-B.inet.0 and VPN-AB.inet.0.
- **vpnab-vpna\_and\_vpnab** installs routes into routing tables VPN-AB.inet.0, VPN-A.inet.0, and VPN-B.inet.0.

Configure the routing table groups:

```
[edit]
routing-options {
  rib-groups {
    vpna-vpnab {
      import-rib [ VPN-A.inet.0 VPN-AB.inet.0 ];
    }
    vpnb-vpnab {
      import-rib [ VPN-B.inet.0 VPN-AB.inet.0 ];
    }
    vpnab-vpna_and_vpnb {
      import-rib [ VPN-AB.inet.0 VPN-A.inet.0 VPN-B.inet.0 ];
    }
  }
}
```

## Configuring Static Routes Between the PE and CE Routers

To configure static routing between the PE1 router and the CE1, CE2, and CE3 routers, you must configure routing instances for VPN A, VPN B, and VPN AB (you configure static routing under each instance):

- [Configuring the Routing Instance for VPN A on page 185](#)
- [Configuring the Routing Instance for VPN AB on page 186](#)
- [Configuring the Routing Instance for VPN B on page 186](#)
- [Configuring VPN Policy on page 187](#)

### Configuring the Routing Instance for VPN A

On Router PE1, configure VPN A:

```
[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      interface-routes {
        rib-group inet vpna-vpnab;
      }
      static {
        route 10.255.14.155/32 next-hop 192.168.197.141;
        route 10.255.14.185/32 next-hop 192.168.197.178;
      }
    }
  }
}
```

The **interface-routes** statement installs VPN A's interface routes into the routing tables defined in the routing table group **vpna-vpnab**.

The **static** statement configures the static routes that are installed in the VPN-A.inet.0 routing table. The first static route is for Router CE1 (VPN A) and the second is for Router CE2 (in VPN AB).

Next hop **192.168.197.178** is not in VPN A. Route **10.255.14.185/32** cannot be installed in VPN-A.inet.0 unless interface routes from routing instance VPN AB are installed in this routing table. Including the **interface-routes** statements in the VPN AB configuration provides this next hop. Similarly, including the **interface-routes** statement in the VPN AB configuration installs **192.168.197.141** into VPN-AB.inet.0.

---

### Configuring the Routing Instance for VPN AB

On Router PE1, configure VPN AB:

```
[edit]
routing instances {
  VPN-AB {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpnab-import;
    vrf-export vpnab-export;
    routing-options {
      interface-routes {
        rib-group vpnab-vpna_and_vpnb;
      }
      static {
        route 10.255.14.185/32 next-hop 192.168.197.178;
        route 10.255.14.155/32 next-hop 192.168.197.141;
        route 10.255.14.186/32 next-hop 192.168.197.242;
      }
    }
  }
}
```

In this configuration, the following static routes are installed in the VPN-AB.inet.0 routing table:

- **10.255.14.185/32** is for Router CE2 (in VPN AB)
- **10.255.14.155/32** is for Router CE1 (in VPN A)
- **10.255.14.186/32** is for Router CE3 (in VPN B)

Next hops **192.168.197.141** and **192.168.197.242** do not belong to VPN AB. Routes **10.255.14.155/32** and **10.255.14.186/32** cannot be installed in VPN-AB.inet.0 unless interface routes from VPN A and VPN B are installed in this routing table. The interface route configurations in VPN A and VPN B routing instances provide these next hops.

---

### Configuring the Routing Instance for VPN B

On Router PE1, configure VPN B:

```
[edit]
routing instances {
  VPN-B {
```

```

instance-type vrf;
interface fe-1/0/2.0;
route-distinguisher 10.255.14.175:10;
vrf-import vpnb-import;
vrf-export vpnb-export;
routing-options {
  interface-routes {
    rib-group inet vpnb-vpnab;
  }
  static {
    route 10.255.14.186/32 next-hop 192.168.197.242;
    route 10.255.14.185/32 next-hop 192.168.197.178;
  }
}
}

```

When you configure the routing instance for VPN B, these static routes are placed in VPNB.inet.0:

- **10.255.14.186/32** is for Router CE3 (in VPN B)
- **10.255.14.185/32** is for Router CE2 (in VPN AB)

Next hop **192.168.197.178** does not belong to VPN B. Route **10.255.14.185/32** cannot be installed in VPN-B.inet.0 unless interface routes from VPN AB are installed in this routing table. The interface route configuration in VPN AB provides this next hop.

### Configuring VPN Policy

The **vrf-import** and **vrf-export** policy statements that you configure for overlapping VPNs are the same as policy statements for regular VPNs, except that you include the **from interface** statement in each VRF export policy. This statement forces each VPN to announce only those routes that originated from that VPN. For example, VPN A has routes that originated in VPN A and VPN AB. If you do not include the **from interface** statement, VPN A announces its own routes as well as VPN AB's routes, so the remote PE router receives multiple announcements for the same routes. Including the **from interface** statement restricts each VPN to announcing only the routes it originated and allows you to filter out the routes imported from other routing tables for local connectivity.

In this configuration example, the **vpnab-import** policy accepts routes from VPN A, VPN B, and VPN AB. The **vpna-export** policy exports only routes that originate in VPN A. Similarly, the **vpnb-export** and **vpnab-export** policies export only routes that originate within the respective VPNs.

On Router PE1, configure the following VPN import and export policies:

```

[edit]
policy-options {
  policy-statement vpna-import {
    term a {
      from {
        protocol bgp;
        community VPNA-comm;
      }
    }
  }
}

```

```
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement vpnb-import {
    term a {
        from {
            protocol bgp;
            community VPNB-comm;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement vpnab-import {
    term a {
        from {
            protocol bgp;
            community [ VPNA-comm VPNB-comm ];
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement vpna-export {
    term a {
        from {
            protocol static;
            interface fe-1/0/0.0;
        }
        then {
            community add VPNA-comm;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement vpnb-export {
    term a {
        from {
            protocol static;
            interface fe-1/0/2.0;
        }
        then {
            community add VPNB-comm;
            accept;
        }
    }
}
```

```

    term b {
        then reject;
    }
}
policy-statement vpnab-export {
    term a {
        from {
            protocol static;
            interface fe-1/1/0.0;
        }
        then {
            community add VPNB-comm;
            community add VPNA-comm;
            accept;
        }
    }
    term b {
        then reject;
    }
}
community VPNA-comm members target:69:1;
community VPNB-comm members target:69:2;
}

```

On Router PE1, apply the VPN import and export policies:

```

[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                rib-group vpna-vpnab;
                route 10.255.14.155/32 next-hop 192.168.197.141;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
        }
    }
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        routing-options {
            static {
                rib-group vpnab-vpna_and_vpnab;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
        }
    }
    VPN-B {

```

```

instance-type vrf;
interface fe-1/0/2.0;
route-distinguisher 10.255.14.175:10;
vrf-import vpnb-import;
vrf-export vpnb-export;
routing-options {
  static {
    rib-group vpnb-vpnab;
    route 10.255.14.186/32 next-hop 192.168.197.242;
  }
}
}
}

```

For VPN A, include the **routing-options** statement at the **[edit routing-instances routing-instance-name]** hierarchy level to install the static routes directly into the routing tables defined in the routing table group **vpna-vpnab**. For VPN AB, the configuration installs the static route directly into the routing tables defined in the routing table group **vpnab-vpna** and **vpnab-vpnb**. For VPN B the configuration installs the static route directly into the routing tables defined in the routing table group **vpnb-vpnab**.

## Configuring BGP Between the PE and CE Routers

In this configuration example, the **vpna-site1** BGP group for VPN A installs the routes learned from the BGP session into the routing tables defined in the **vpna-vpnab** routing table group. For VPN AB, the **vpnab-site1** group installs the routes learned from the BGP session into the routing tables defined in the **vpnab-vpna\_and\_vpnb** routing table group. For VPN B, the **vpnb-site1** group installs the routes learned from the BGP session into the routing tables defined in the **vpnb-vpnab** routing table group. Interface routes are not needed for this configuration.

The VRF import and export policies are similar to those defined in “[Configuring Static Routes Between the PE and CE Routers](#)” on page 185, except the export protocol is BGP instead of a static route. On all **vrf-export** policies, you use the **from protocol bgp** statement.

On Router PE1, configure BGP between the PE and CE routers:

```

[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
      bgp {
        group vpna-site1 {
          family inet {
            unicast {
              rib-group vpna-vpnab;
            }
          }
        }
      }
    }
  }
}

```

```

        peer-as 1;
        neighbor 192.168.197.141;
    }
}
}
VPN-AB {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpnab-import;
    vrf-export vpnab-export;
    protocols {
        bgp {
            group vpnab-site1 {
                family inet {
                    unicast {
                        rib-group vpnab-vpna_and_vpnb;
                    }
                }
            }
            peer-as 9;
            neighbor 192.168.197.178;
        }
    }
}
VPN-B {
    instance-type vrf;
    interface fe-1/0/2.0;
    route-distinguisher 10.255.14.175:10;
    vrf-import vpnb-import;
    vrf-export vpnb-export;
    protocols {
        bgp {
            group vpnb-site1 {
                family inet {
                    unicast {
                        rib-group vpnb-vpnab;
                    }
                }
            }
            neighbor 192.168.197.242 {
                peer-as 10;
            }
        }
    }
}
}

```

## Configuring OSPF Between the PE and CE Routers

In this configuration example, routes learned from the OSPF session for VPN A are installed into the routing tables defined in the **vpna-vpnab** routing table group. For VPN AB, routes learned from the OSPF session are installed into the routing tables defined in the

**vpnab-vpna\_and\_vpnb** routing table group. For VPN B, routes learned from the OSPF session are installed into the routing tables defined in the **vpnb-vpnab** routing table group.

The VRF import and export policies are similar to those defined in “[Configuring Static Routes Between the PE and CE Routers](#)” on page 185 and “[Configuring BGP Between the PE and CE Routers](#)” on page 190, except the export protocol is OSPF instead of BGP or a static route. Therefore, on all **vrf-export** policies, you use the **from protocol ospf** statement instead of the **from protocol <static | bgp>** statement.

On Router PE1, configure OSPF between the PE and CE routers:

```
[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
      ospf {
        rib-group vpna-vpnab;
        export vpna-import;
        area 0.0.0.0 {
          interface fe-1/0/0.0;
        }
      }
    }
  }
  VPN-AB {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpnab-import;
    vrf-export vpnab-export;
    protocols {
      ospf {
        rib-group vpnab-vpna_and_vpnb;
        export vpnab-import;
        area 0.0.0.0 {
          interface fe-1/1/0.0;
        }
      }
    }
  }
  VPN-B {
    instance-type vrf;
    interface fe-1/0/2.0;
    route-distinguisher 10.255.14.175:10;
    vrf-import vpnb-import;
    vrf-export vpnb-export;
    protocols {
      ospf {
        rib-group vpnb-vpnab;
        export vpnb-import;
      }
    }
  }
}
```

```

        area 0.0.0.0 {
            interface fe-1/0/2.0;
        }
    }
}
}

```

## Configuring Static, BGP, and OSPF Routes Between PE and CE Routers

This section shows how to configure the routes between the PE and CE routers by using a combination of static routes, BGP, and OSPF:

- The connection between Router PE1 and Router CE1 uses static routing.
- The connection between Router PE1 and Router CE2 uses BGP.
- The connection between Router PE1 and Router CE3 uses OSPF.

Here, the configuration for VPN AB also includes a static route to CE1.

On Router PE1, configure a combination of static routing, BGP, and OSPF between the PE and CE routers:

```

[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      static {
        rib-group vpna-vpnab;
        route 10.255.14.155/32 next-hop 192.168.197.141;
      }
    }
  }
  VPN-AB {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpnab-import;
    vrf-export vpnab-export;
    protocols {
      bgp {
        group vpnab-site1 {
          family inet {
            unicast {
              rib-group vpnab-vpna_and_vpnab;
            }
          }
        }
        export to-vpnab-site1;
        peer-as 9;
        neighbor 192.168.197.178;
      }
    }
  }
}

```

```
    }  
  }  
}  
VPN-B {  
  instance-type vrf;  
  interface fe-1/0/2.0;  
  route-distinguisher 10.255.14.175:10;  
  vrf-import vpnb-import;  
  vrf-export vpnb-export;  
  protocols {  
    ospf {  
      rib-group vpnb-vpnab;  
      export vpnb-import;  
      area 0.0.0.1 {  
        interface t3-0/3/3.0;  
      }  
    }  
  }  
}  
}  
policy-options {  
  policy-statement to-vpnab-site1 {  
    term a {  
      from protocol static;  
      then accept;  
    }  
    term b {  
      from protocol bgp;  
      then accept;  
    }  
    term c {  
      then reject;  
    }  
  }  
}
```

---

## Configuring Overlapping VPNs Using Automatic Route Export

A problem with multiple routing instances is how to export routes between routing instances. You can accomplish this in Junos OS by configuring routing table groups for each routing instance that needs to export routes to other routing tables. For information about how to configure overlapping VPNs by using routing table groups, see [“Configuring Overlapping VPNs Using Routing Table Groups” on page 183](#).

However, using routing table groups has limitations:

- Routing table group configuration is complex. You must define a unique routing table group for each routing instance that will export routes.
- You must also configure a unique routing table group for each protocol that will export routes.

To limit and sometimes eliminate the need to configure routing table groups in multiple routing instance topologies, you can use the functionality provided by the **auto-export** statement.

The **auto-export** statement is particularly useful for configuring overlapping VPNs—VPN configurations where more than one VRF routing instance lists the same community route target in its **vrf-import** policy. The **auto-export** statement finds out which routing tables to export routes from and import routes to by examining the existing policy configuration.

The **auto-export** statement automatically exports routes between the routing instances referencing a given route target community. When the **auto-export** statement is configured, a VRF target tree is constructed based on the **vrf-import** and **vrf-export** policies configured on the system. If a routing instance references a route target in its **vrf-import** policy, the route target is added to the import list for the target. If it references a specific route target in its **vrf-export** policy, the route target is added to the export list for that target. Route targets where there is a single importer that matches a single exporter or with no importers or exporters are ignored.

Changes to routing tables that export route targets are tracked. When a route change occurs, the routing instance's **vpn-export** policy is applied to the route. If it is allowed, the route is imported to all the import tables (subject to the **vrf-import** policy) of the route targets set by the export policy.

The sections that follow describe how to configure overlapping VPNs by using the **auto-export** statement for inter-instance export in addition to routing table groups:

- [Configuring Overlapping VPNs with BGP and Automatic Route Export on page 195](#)
- [Configuring Overlapping VPNs and Additional Tables on page 196](#)
- [Configuring Automatic Route Export for All VRF Instances on page 197](#)

## Configuring Overlapping VPNs with BGP and Automatic Route Export

The following example provides the configuration for an overlapping VPN where BGP is used between the PE and CE routers.

Configure routing instance **VPN-A**:

```
[edit]
routing-instances {
  VPN-A {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.175:3;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      auto-export;
    }
    protocols {
      bgp {
        group vpna-site1 {
```

```

        peer-as 1;
        neighbor 192.168.197.141;
    }
}
}
}

```

Configure routing instance **VPN-AB**:

```

[edit]
routing-instances {
  VPN-AB {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpnab-import;
    vrf-export vpnab-export;
    routing-options {
      auto-export;
    }
    protocols {
      bgp {
        group vpnab-site1 {
          peer-as 9;
          neighbor 192.168.197.178;
        }
      }
    }
  }
}

```

For this configuration, the **auto-export** statement replaces the functionality that was provided by a routing table group configuration. However, sometimes additional configuration is required.

Since the **vrf-import** policy and the **vrf-export** policy from which the **auto-export** statement deduces the import and export matrix are configured on a per-instance basis, you must be able to enable or disable them for unicast and multicast, in case multicast network layer reachability information (NLRI) is configured.

## Configuring Overlapping VPNs and Additional Tables

You might need to use the **auto-export** statement between overlapping VPNs but require that a subset of the routes learned from a VRF table be installed into the inet.0 table or in routing-instance.inet.2.

To support this type of scenario, where not all of the information needed is present in the **vrf-import** and **vrf-export** policies, you configure an additional list of routing tables by using an additional routing table group.

To add routes from **VPN-A** and **VPN-AB** to inet.0 in the example described, you need to include the following additional configuration statements:

Configure the routing options:

```
[edit]
routing-options {
  rib-groups {
    inet-access {
      import-rib inet.0;
    }
  }
}
```

Configure routing instance **VPN-A**:

```
[edit]
routing-instances {
  VPN-A {
    routing-options {
      auto-export {
        family inet {
          unicast {
            rib-group inet-access;
          }
        }
      }
    }
  }
}
```

Configure routing instance **VPN-AB**:

```
[edit]
routing-instances {
  VPN-AB {
    routing-options {
      auto-export {
        family inet {
          unicast {
            rib-group inet-access;
          }
        }
      }
    }
  }
}
```

Routing table groups are used in this configuration differently from how they are generally used in Junos OS. Routing table groups normally require that the exporting routing table be referenced as the primary import routing table in the routing table group. For this configuration, the restriction does not apply. The routing table group functions as an additional list of tables to which to export routes.

## Configuring Automatic Route Export for All VRF Instances

The following configuration allows you to configure the **auto-export** statement for all of the routing instances in a configuration group:

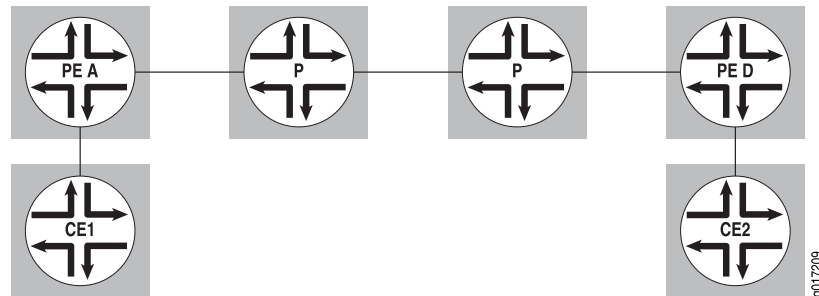
```
[edit]
groups {
  vrf-export-on {
```

```
routing-instances {  
    <*> {  
        routing-options {  
            auto-export;  
        }  
    }  
}  
}  
}  
} apply-groups vrf-export-on;
```

## Configuring a GRE Tunnel Interface Between PE Routers

This example shows how to configure a generic routing encapsulation (GRE) tunnel interface between PE routers to provide VPN connectivity. You can use this configuration to tunnel VPN traffic across a non-MPLS core network. The network topology used in this example is shown in [Figure 29 on page 198](#). The P routers shown in this illustration do not run MPLS.

Figure 29: PE Routers A and D Connected by a GRE Tunnel Interface



For configuration information, see the following sections:

- [Configuring the Routing Instance on Router A on page 198](#)
- [Configuring the Routing Instance on Router D on page 199](#)
- [Configuring MPLS, BGP, and OSPF on Router A on page 199](#)
- [Configuring MPLS, BGP, and OSPF on Router D on page 200](#)
- [Configuring the Tunnel Interface on Router A on page 200](#)
- [Configuring the Tunnel Interface on Router D on page 201](#)
- [Configuring the Routing Options on Router A on page 201](#)
- [Configuring the Routing Options on Router D on page 201](#)
- [Configuration Summary for Router A on page 202](#)
- [Configuration Summary for Router D on page 203](#)

## Configuring the Routing Instance on Router A

Configure a routing instance on Router A:

```
[edit routing-instances]
gre-config {
```

```

instance-type vrf;
interface fe-1/0/0.0;
route-distinguisher 10.255.14.176:69;
vrf-import import-config;
vrf-export export-config;
protocols {
  ospf {
    export import-config;
    area 0.0.0.0 {
      interface all;
    }
  }
}
}

```

### Configuring the Routing Instance on Router D

Configure a routing instance on Router D:

```

[edit routing-instances]
gre-config {
  instance-type vrf;
  interface fe-1/0/1.0;
  route-distinguisher 10.255.14.178:69;
  vrf-import import-config;
  vrf-export export-config;
  protocols {
    ospf {
      export import-config;
      area 0.0.0.0 {
        interface all;
      }
    }
  }
}
}

```

### Configuring MPLS, BGP, and OSPF on Router A

Although you do not need to configure MPLS on the P routers in this example, it is needed on the PE routers for the interface between the PE and CE routers and on the GRE interface (**gr-1/1/0.0**) linking the PE routers (Router A and Router D). Configure MPLS, BGP, and OSPF on Router A:

```

[edit protocols]
mpls {
  interface all;
}
bgp {
  group pe-to-pe {
    type internal;
    neighbor 10.255.14.178 {
      family inet-vpn {
        unicast;
      }
    }
  }
}

```

```
}
ospf {
  area 0.0.0.0 {
    interface all;
    interface gr-1/1/0.0 {
      disable;
    }
  }
}
```

## Configuring MPLS, BGP, and OSPF on Router D

Although you do not need to configure MPLS on the P routers in this example, it is needed on the PE routers for the interface between the PE and CE routers and on the GRE interface (**gr-1/1/0.0**) linking the PE routers (Router D and Router A). Configure MPLS, BGP, and OSPF on Router D:

```
[edit protocols]
mpls {
  interface all;
}
bgp {
  group pe-to-pe {
    type internal;
    neighbor 10.255.14.176 {
      family inet-vpn {
        unicast;
      }
    }
  }
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface all;
    interface fxp0.0 {
      disable;
    }
    interface gr-1/1/0.0 {
      disable;
    }
  }
}
```

## Configuring the Tunnel Interface on Router A

Configure the tunnel interface on Router A (the tunnel is unnumbered):

```
[edit interfaces interface-name]
unit 0 {
  tunnel {
    source 10.255.14.176;
    destination 10.255.14.178;
  }
  family inet;
  family mpls;
```

```
}
```

## Configuring the Tunnel Interface on Router D

Configure the tunnel interface on Router D (the tunnel is unnumbered):

```
[edit interfaces interface-name]
unit 0 {
  tunnel {
    source 10.255.14.178;
    destination 10.255.14.176;
  }
  family inet;
  family mpls;
}
```

## Configuring the Routing Options on Router A

As part of the routing options configuration for Router A, you need to configure routing table groups to enable VPN route resolution in the inet.3 routing table.

Configure the routing options on Router A:

```
[edit routing-options]
interface-routes {
  rib-group inet if-rib;
}
rib inet.3 {
  static {
    route 10.255.14.178/32 next-hop gr-1/1/0.0;
  }
}
rib-groups {
  if-rib {
    import-rib [ inet.0 inet.3 ];
  }
}
```

## Configuring the Routing Options on Router D

As part of the routing options configuration for Router D, you need to configure routing table groups to enable VPN route resolution in the inet.3 routing table.

Configure the routing options on Router D:

```
[edit routing-options]
interface-routes {
  rib-group inet if-rib;
}
rib inet.3 {
  static {
    route 10.255.14.176/32 next-hop gr-1/1/0.0;
  }
}
rib-groups {
  if-rib {
```

```

        import-rib [ inet.0 inet.3 ];
    }
}

```

## Configuration Summary for Router A

Configure the Routing Instance	<pre> gre-config {   instance-type vrf;   interface fe-1/0/0.0;   route-distinguisher 10.255.14.176:69;   vrf-import import-config;   vrf-export export-config;   protocols {     ospf {       export import-config;       area 0.0.0.0 {         interface all;       }     }   } } </pre>
Configure MPLS	<pre> mpls {   interface all; } </pre>
Configure BGP	<pre> bgp {   traceoptions {     file bgp.trace world-readable;     flag update detail;   }   group pe-to-pe {     type internal;     neighbor 10.255.14.178 {       family inet-vpn {         unicast;       }     }   } } </pre>
Configure OSPF	<pre> ospf {   area 0.0.0.0 {     interface all;     interface gr-1/1/0.0 {       disable;     }   } } </pre>
Configure the Tunnel Interface	<pre> interface-name {   unit 0 {     tunnel {       source 10.255.14.176;       destination 10.255.14.178;     }   } } </pre>

```

    }
    family inet;
    family mpls;
  }
}

Configure Routing Options interface-routes {
    rib-group inet if-rib;
}
rib inet.3 {
    static {
        route 10.255.14.178/32 next-hop gr-1/1/0.0;
    }
}
rib-groups {
    if-rib {
        import-rib [ inet.0 inet.3 ];
    }
}

```

### Configuration Summary for Router D

<b>Configure the Routing Instance</b>	<pre> gre-config {     instance-type vrf;     interface fe-1/0/1.0;     route-distinguisher 10.255.14.178:69;     vrf-import import-config;     vrf-export export-config;     protocols {         ospf {             export import-config;             area 0.0.0.0 {                 interface all;             }         }     } } </pre>
<b>Configure MPLS</b>	<pre> mpls {     interface all; } </pre>
<b>Configure BGP</b>	<pre> bgp {     group pe-to-pe {         type internal;         neighbor 10.255.14.176 {             family inet-vpn {                 unicast;             }         }     } } </pre>
<b>Configure OSPF</b>	<pre> ospf {     traffic-engineering; } </pre>

```

area 0.0.0.0 {
  interface all;
  interface fxp0.0 {
    disable;
  }
  interface gr-1/1/0.0 {
    disable;
  }
}
}

Configure the Tunnel
Interface
interface-name {
  unit 0 {
    tunnel {
      source 10.255.14.178;
      destination 10.255.14.176;
    }
    family inet;
    family mpls;
  }
}

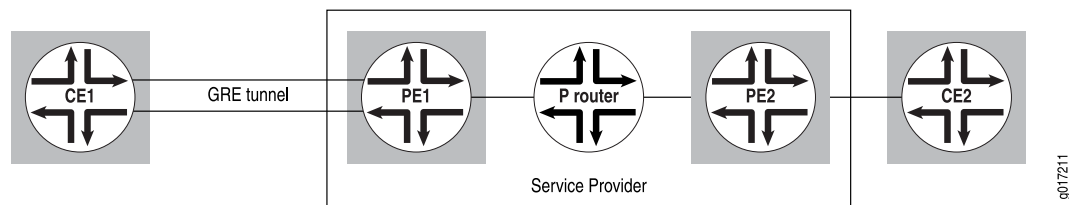
Configure the Routing
Options
interface-routes {
  rib-group inet if-rib;
}
rib inet.3 {
  static {
    route 10.255.14.176/32 next-hop gr-1/1/0.0;
  }
}
rib-groups {
  if-rib {
    import-rib [ inet.0 inet.3 ];
  }
}

```

## Configuring a GRE Tunnel Interface Between a PE and CE Router

This example shows how to configure a GRE tunnel interface between a PE router and a CE router. You can use this configuration to tunnel VPN traffic across a non-MPLS core network. The network topology used in this example is shown in [Figure 30 on page 204](#).

Figure 30: GRE Tunnel Between the CE Router and the PE Router



For this example, complete the procedures described in the following sections:

- [Configuring the Routing Instance Without the Encapsulating Interface on page 205](#)
- [Configuring the Routing Instance with the Encapsulating Interface on page 206](#)
- [Configuring the GRE Tunnel Interface on Router CE1 on page 208](#)

## Configuring the Routing Instance Without the Encapsulating Interface

You can configure the routing instance either with or without the encapsulating interface. The following sections explain how to configure the routing instance without it:

- [Configuring the Routing Instance on Router PE1 on page 205](#)
- [Configuring the GRE Tunnel Interface on Router PE1 on page 205](#)
- [Configuring the Encapsulation Interface on Router PE1 on page 206](#)

### Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
  instance-type vrf;
  interface gr-1/2/0.0;
  route-distinguisher 10.255.14.174:1;
  vrf-import vpna-import;
  vrf-export vpna-export;
  protocols {
    bgp {
      group vpna {
        type external;
        peer-as 100;
        as-override;
        neighbor 10.49.2.1;
      }
    }
  }
}
```

### Configuring the GRE Tunnel Interface on Router PE1

Configure the GRE tunnel interface on Router PE1:

```
[edit interfaces gr-1/2/0]
unit 0 {
  tunnel {
    source 192.168.197.249;
    destination 192.168.197.250;
  }
  family inet {
    address 10.49.2.2/30;
  }
}
```

In this example, interface **t3-0/1/3** acts as the encapsulating interface for the GRE tunnel.

When you configure the **clear-dont-fragment-bit** statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```
user@host# show interfaces gr-1/2/0
unit 0 {
  clear-dont-fragment-bit;
  family inet {
    mtu 9100;
    address 10.10.1.1/32;
  }
  family mpls {
    mtu 9100;
  }
}
```

---

### Configuring the Encapsulation Interface on Router PE1

Configure the encapsulation interface on Router PE1:

```
[edit interfaces t3-0/1/3]
unit 0 {
  family inet {
    address 192.168.197.249/30;
  }
}
```

### Configuring the Routing Instance with the Encapsulating Interface

If the tunnel-encapsulating interface, **t3-0/1/3**, is also configured under the routing instance, then you need to specify the name of that routing instance under the interface definition. The system uses this routing instance to search for the tunnel destination address.

To configure the routing instance with the encapsulating interface, you perform the steps in the following sections:

- [Configuring the Routing Instance on Router PE1 on page 206](#)
- [Configuring the GRE Tunnel Interface on Router PE1 on page 207](#)
- [Configuring the Encapsulation Interface on Router PE1 on page 208](#)

---

### Configuring the Routing Instance on Router PE1

If you configure the tunnel-encapsulating interface under the routing instance, then configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
  instance-type vrf;
  interface gr-1/2/0.0;
  interface t3-0/1/3.0;
  route-distinguisher 10.255.14.174:1;
  vrf-import vpna-import;
```

```

vrf-export vpna-export;
protocols {
  bgp {
    group vpna {
      type external;
      peer-as 100;
      as-override;
      neighbor 10.49.2.1;
    }
  }
}

```

### Configuring the GRE Tunnel Interface on Router PE1

Configure the GRE tunnel interface on Router PE1:

```

[edit interfaces gr-1/2/0]
unit 0 {
  tunnel {
    source 192.168.197.249;
    destination 192.168.197.250;
    routing-instance {
      destination vpna;
    }
  }
  family inet {
    address 10.49.2.2/30;
  }
}

```

When you configure the **clear-dont-fragment-bit** statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```

user@host# show interfaces gr-1/2/0
unit 0 {
  clear-dont-fragment-bit;
  family inet {
    mtu 9100;
    address 10.10.1.1/32;
  }
  family mpls {
    mtu 9100;
  }
}

```

When you configure the **clear-dont-fragment-bit** statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```

user@host# show interfaces gr-1/2/0
unit 0 {

```

```

clear-dont-fragment-bit;
family inet {
    mtu 9100;
    address 10.10.1.1/32;
}
family mpls {
    mtu 9100;
}
}

```

### Configuring the Encapsulation Interface on Router PE1

Configure the encapsulation interface on Router PE1:

```

[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}

```

### Configuring the GRE Tunnel Interface on Router CE1

Configure the GRE tunnel interface on Router CE1:

```

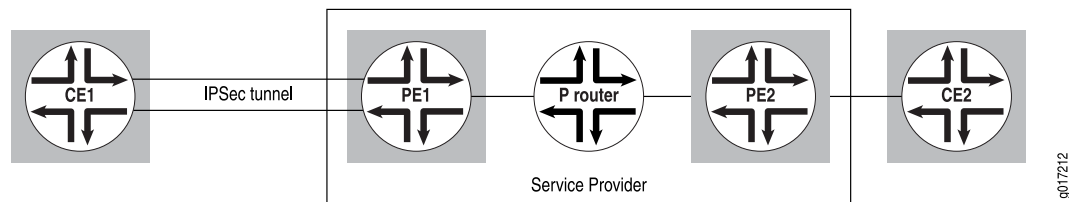
[edit interfaces gr-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.250;
        destination 192.168.197.249;
    }
    family inet {
        address 10.49.2.1/30;
    }
}

```

### Configuring an ES Tunnel Interface Between a PE and CE Router

This example shows how to configure an ES tunnel interface between a PE router and a CE router in a Layer 3 VPN. The network topology used in this example is shown in [Figure 31 on page 208](#).

Figure 31: ES Tunnel Interface (IPsec Tunnel)



To configure this example, you perform the steps in the following sections:

- [Configuring IPsec on Router PE1 on page 209](#)
- [Configuring the Routing Instance Without the Encapsulating Interface on page 209](#)

- [Configuring the Routing Instance with the Encapsulating Interface on page 210](#)
- [Configuring the ES Tunnel Interface on Router CE1 on page 211](#)
- [Configuring IPsec on Router CE1 on page 212](#)

## Configuring IPsec on Router PE1

Configure IP Security (IPsec) on Router PE1:

```
[edit security]
ipsec {
  security-association sa-esp-manual {
    mode tunnel;
    manual {
      direction bidirectional {
        protocol esp;
        spi 16000;
        authentication {
          algorithm hmac-md5-96;
          key ascii-text
            "$9$ABULt1heK87dsWLDk.P3nrevM7V24ZHkPaZ/tpOcSvWLNwgZUH";
        }
        encryption {
          algorithm des-cbc;
          key ascii-text "$9$/H8Q90IyrvL7VKMZjHqQzcyleLN";
        }
      }
    }
  }
}
```

## Configuring the Routing Instance Without the Encapsulating Interface

You can configure the routing instance on Router PE1 with or without the encapsulating interface (t3-0/1/3 in this example). The following sections explain how to configure the routing instance without it:

- [Configuring the Routing Instance on Router PE1 on page 209](#)
- [Configuring the ES Tunnel Interface on Router PE1 on page 210](#)
- [Configuring the Encapsulating Interface for the ES Tunnel on page 210](#)

## Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
  instance-type vrf;
  interface es-1/2/0.0;
  route-distinguisher 10.255.14.174:1;
  vrf-import vpna-import;
  vrf-export vpna-export;
  protocols {
    bgp {
      group vpna {
```

```
        type external;
        peer-as 100;
        as-override;
        neighbor 10.49.2.1;
    }
}
}
```

---

### Configuring the ES Tunnel Interface on Router PE1

Configure the ES tunnel interface on Router PE1:

```
[edit interfaces es-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.249;
        destination 192.168.197.250;
    }
    family inet {
        address 10.49.2.2/30;
        ipsec-sa sa-esp-manual;
    }
}
```

---

### Configuring the Encapsulating Interface for the ES Tunnel

For this example, interface **t3-0/1/3** is the encapsulating interface for the ES tunnel. Configure interface **t3-0/1/3**:

```
[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}
```

### Configuring the Routing Instance with the Encapsulating Interface

If the tunnel-encapsulating interface, **t3-0/1/3**, is also configured under the routing instance, you need to specify the routing instance name under the interface definition. The system uses this routing instance to search for the tunnel destination address for the IPsec tunnel using manual security association.

The following sections explain how to configure the routing instance with the encapsulating interface:

- [Configuring the Routing Instance on Router PE1 on page 210](#)
- [Configuring the ES Tunnel Interface on Router PE1 on page 211](#)
- [Configuring the Encapsulating Interface on Router PE1 on page 211](#)

---

### Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1 (including the tunnel encapsulating interface):

```
[edit routing-instances]
vpna {
  instance-type vrf;
  interface es-1/2/0.0;
  interface t3-0/1/3.0;
  route-distinguisher 10.255.14.174:1;
  vrf-import vpna-import;
  vrf-export vpna-export;
  protocols {
    bgp {
      group vpna {
        type external;
        peer-as 100;
        as-override;
        neighbor 10.49.2.1;
      }
    }
  }
}
```

### Configuring the ES Tunnel Interface on Router PE1

Configure the ES tunnel interface on Router PE1:

```
[edit interfaces es-1/2/0]
unit 0 {
  tunnel {
    source 192.168.197.249;
    destination 192.168.197.250;
    routing-instance {
      destination vpna;
    }
  }
  family inet {
    address 10.49.2.2/30;
    ipsec-sa sa-esp-manual;
  }
}
```

### Configuring the Encapsulating Interface on Router PE1

Configure the encapsulating interface on Router PE1:

```
[edit interfaces t3-0/1/3]
unit 0 {
  family inet {
    address 192.168.197.249/30;
  }
}
```

### Configuring the ES Tunnel Interface on Router CE1

Configure the ES tunnel interface on Router CE1:

```
[edit interfaces es-1/2/0]
unit 0 {
  tunnel {
```

```
        source 192.168.197.250;
        destination 192.168.197.249;
    }
    family inet {
        address 10.49.2.1/30;
        ipsec-sa sa-esp-manual;
    }
}
```

## Configuring IPsec on Router CE1

Configure IPsec on Router CE1:

```
[edit security]
ipsec {
    security-association sa-esp-manual {
        mode tunnel;
        manual {
            direction bidirectional {
                protocol esp;
                spi 16000;
                authentication {
                    algorithm hmac-md5-96;
                    key ascii-text
                        "$9$ABULt1heK87dsWLDk.P3nrevM7V24ZHkPaZ/tp0cSvWLNwgZUH";
                }
                encryption {
                    algorithm des-cbc;
                    key ascii-text "$9$/H8Q90IyrvL7VKMZjHqQzcyleLN";
                }
            }
        }
    }
}
```

## Layer 3 VPN Load Balancing Using IP Header Filtering

---

- [Layer 3 VPN Load Balancing Overview on page 212](#)
- [Example: Load Balancing Layer 3 VPN Traffic While Simultaneously Using IP Header Filtering on page 213](#)

### Layer 3 VPN Load Balancing Overview

The load balancing feature allows a device to divide incoming and outgoing traffic along multiple paths in order to reduce congestion in the network. Load balancing improves the utilization of various network paths, and provides more effective network bandwidth.

When multiple protocols are in use, the device uses the route preference value (also known as the administrative distance value) to select a route. While using a single routing protocol, the router chooses the path with the lowest cost (or metric) to the destination. If the device receives and installs multiple paths with the same route preference and same cost to a destination, load balancing must be configured.

In a network with both internal and external BGP paths installed among devices in different autonomous systems, BGP selects only a single best path by default, and does not perform load balancing. A Layer 3 VPN with internal and external BGP paths uses the **multipath** statement for protocol-independent load balancing. When you include the **multipath** statement in a routing instance, protocol-independent load balancing is applied to the default routing table for that routing instance. By using the **vpn-unequal-cost** statement, protocol-independent load balancing is applied to VPN routes. By using the **equal-external-internal** statement, protocol-independent load balancing is applied to both internal and external BGP paths and can be configured in conjunction with IP header filtering (enabled with the **vrf-table-label** statement).

### Example: Load Balancing Layer 3 VPN Traffic While Simultaneously Using IP Header Filtering

This example shows how to configure load balancing in a Layer 3 VPN (with internal and external BGP paths) while simultaneously using IP header filtering.

- [Requirements on page 213](#)
- [Overview on page 213](#)
- [Configuration on page 215](#)
- [Verification on page 223](#)

#### Requirements

This example requires the following hardware and software components:

- M Series Multiservice Edge Routers (M120 and M320 only), MX Series 3D Universal Edge Routers, T Series Core Routers, or PTX Series Transport Routers.
- Junos OS Release 12.1 or later

#### Overview

The following example shows how to configure load balancing while simultaneously using IP header filtering in a Layer 3 VPN.



**NOTE:** This example demonstrates how load balancing and IP header filtering work together. The testing of IP header filtering is out of the scope of this example.

The Junos OS BGP provides a multipath feature that allows load balancing between peers in the same or different autonomous systems (ASs). This example uses the **equal-external-internal** statement at the **[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]** hierarchy level to perform load balancing. The **vrf-table-label** statement is configured at the **[edit routing-instances instance-name]** hierarchy level to enable IP header filtering.

```
[edit]
routing-instances {
  instance-name {
    vrf-table-label;
```

```

routing-options {
  multipath {
    vpn-unequal-cost {
      equal-external-internal;
    }
  }
}

```



**NOTE:** These statements are available only in the context of a routing instance.

In this example, Device CE1 is in AS1 and connected to Device PE1. Devices PE1, PE2, PE3, and P are in AS2. Device CE2 is connected to Devices PE2 and PE3 and is in AS3. Device CE3 is connected to Device PE3 and is in AS4. BGP and MPLS are configured through the network. OSPF is the interior gateway protocol (IGP) that is used in this network.

The configuration for Devices PE1, PE2, and PE3 includes the **equal-external-internal** statement at the **[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]** hierarchy level to enable load balancing in the network. IP header filtering is enabled when the **vrf-table-label** statement is configured at the **[edit routing-instances instance-name]** hierarchy level on the PE devices.

Figure 32 on page 214 shows the topology used in this example.

**Figure 32: Layer 3 VPN Load Balancing Using IP Header Filtering**

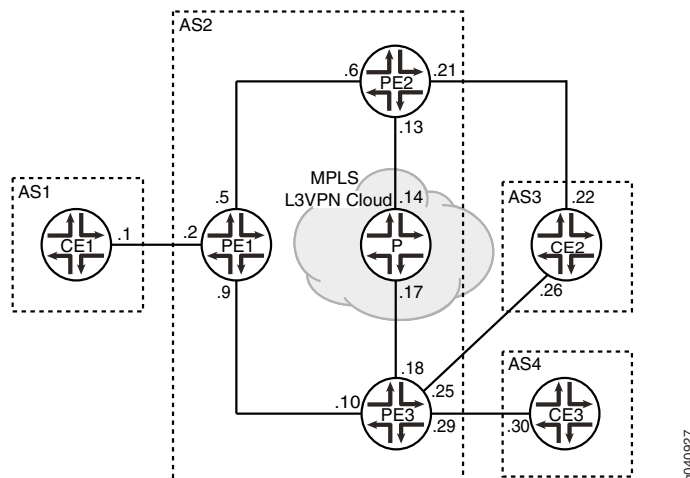


Table 10 on page 215 shows the list of IP addresses used in this example for quick reference.

Table 10: Device IP Address Quick Reference

Device	AS	Device ID	Device Interface Units	Device Interface Unit IPs
CE1	1	1.1.1.1/32	Unit 1	10.1.1.1/30
PE1	2	1.1.1.2/32	Unit 2	10.1.1.2/30
			Unit 5	10.1.2.5/30
			Unit 9	10.1.3.9/30
PE2	2	1.1.1.3/32	Unit 6	10.1.2.6/30
			Unit 13	10.1.4.13/30
			Unit 21	10.1.6.21/30
PE3	2	1.1.1.4/32	Unit 10	10.1.3.10/30
			Unit 18	10.1.5.18/30
			Unit 25	10.1.7.25/30
			Unit 29	10.1.8.29/30
P	2	1.1.1.5/32	Unit 14	10.1.4.14/30
			Unit 17	10.1.5.17/30
CE2	3	1.1.1.6/32	Unit 22	10.1.6.22/30
			Unit 26	10.1.7.26/30
CE3	4	1.1.1.7/32	Unit 30	10.1.8.30/30



**NOTE:** This example was tested using logical systems (logical routers). Therefore all the physical interfaces in the example are the same and the configuration is done on separate logical interfaces. In an non-test network, you will use separate physical routers and separate physical interfaces for the connections to other devices.

### Configuration

#### CLI Quick Configuration

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

**Device CE1**

```
set interfaces ge-2/1/10 unit 1 family inet address 10.1.1.1/30
set interfaces ge-2/1/10 unit 1 family mpls
set interfaces ge-2/1/10 unit 1 description toPE1
set interfaces lo0 unit 4 family inet address 1.1.1.1/32
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 1
set protocols bgp group toPE1 type external
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 peer-as 2
set protocols bgp group toPE1 neighbor 10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

**Device PE1**

```
set interfaces ge-2/1/10 unit 2 family inet address 10.1.1.2/30
set interfaces ge-2/1/10 unit 2 family mpls
set interfaces ge-2/1/10 unit 2 description toCE1
set interfaces ge-2/1/10 unit 5 family inet address 10.1.2.5/30
set interfaces ge-2/1/10 unit 5 family mpls
set interfaces ge-2/1/10 unit 5 description toPE2
set interfaces ge-2/1/10 unit 9 family inet address 10.1.3.9/30
set interfaces ge-2/1/10 unit 9 family mpls
set interfaces ge-2/1/10 unit 9 description toPE3
set interfaces lo0 unit 5 family inet address 1.1.1.2/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface ge-2/1/10.5 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/1/10.9 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal local-address 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.2
set routing-options autonomous-system 2
set routing-options forwarding-table export lb
set routing-instances purple instance-type vrf
set routing-instances purple interface ge-2/1/10.2
set routing-instances purple route-distinguisher 2:1
set routing-instances purple vrf-target target:2:1
set routing-instances purple vrf-table-label
set routing-instances purple protocols bgp group toCE1 type external
set routing-instances purple protocols bgp group toCE1 peer-as 1
set routing-instances purple protocols bgp group toCE1 neighbor 10.1.1.1
set routing-instances purple routing-options multipath vpn-unequal-cost
    equal-external-internal
set policy-options policy-statement lb then load-balance per-packet
```

**Device PE2**

```
set interfaces ge-2/1/10 unit 6 family inet address 10.1.2.6/30
set interfaces ge-2/1/10 unit 6 family mpls
set interfaces ge-2/1/10 unit 6 description toPE1
```

```

set interfaces ge-2/1/10 unit 13 family inet address 10.1.4.13/30
set interfaces ge-2/1/10 unit 13 family mpls
set interfaces ge-2/1/10 unit 13 description toP
set interfaces ge-2/1/10 unit 21 family inet address 10.1.6.21/30
set interfaces ge-2/1/10 unit 21 family mpls
set interfaces ge-2/1/10 unit 21 description toCE2
set interfaces lo0 unit 6 family inet address 1.1.1.3/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.6 passive
set protocols ospf area 0.0.0.0 interface ge-2/1/10.6 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/1/10.13 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal local-address 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.3
set routing-options autonomous-system 2
set routing-options forwarding-table export lb
set routing-instances purple instance-type vrf
set routing-instances purple interface ge-2/1/10.21
set routing-instances purple route-distinguisher 2:1
set routing-instances purple vrf-target target:2:1
set routing-instances purple vrf-table-label
set routing-instances purple protocols bgp group toCE2 type external
set routing-instances purple protocols bgp group toCE2 peer-as 3
set routing-instances purple protocols bgp group toCE2 neighbor 10.1.6.22
set routing-instances purple routing-options multipath vpn-unequal-cost
    equal-external-internal
set policy-options policy-statement lb then load-balance per-packet

```

#### Device PE3

```

set interfaces ge-2/1/10 unit 10 family inet address 10.1.3.10/30
set interfaces ge-2/1/10 unit 10 family mpls
set interfaces ge-2/1/10 unit 10 description toPE1
set interfaces ge-2/1/10 unit 18 family inet address 10.1.5.18/30
set interfaces ge-2/1/10 unit 18 family mpls
set interfaces ge-2/1/10 unit 18 description toP
set interfaces ge-2/1/10 unit 25 family inet address 10.1.7.25/30
set interfaces ge-2/1/10 unit 25 family mpls
set interfaces ge-2/1/10 unit 25 description toCE2
set interfaces ge-2/1/10 unit 29 family inet address 10.1.8.29/30
set interfaces ge-2/1/10 unit 29 family mpls
set interfaces ge-2/1/10 unit 29 description toCE3
set interfaces lo0 unit 7 family inet address 1.1.1.4/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.7 passive
set protocols ospf area 0.0.0.0 interface ge-2/1/10.10 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/1/10.18 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal local-address 1.1.1.4
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family route-target

```

```
set protocols bgp group toInternal neighbor 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.3
set routing-options router-id 1.1.1.4
set routing-options autonomous-system 2
set routing-options forwarding-table export lb
set routing-instances purple instance-type vrf
set routing-instances purple interface ge-2/1/10.25
set routing-instances purple interface ge-2/1/10.29
set routing-instances purple route-distinguisher 2:1
set routing-instances purple vrf-target target:2:1
set routing-instances purple vrf-table-label
set routing-instances purple protocols bgp group toCE2 type external
set routing-instances purple protocols bgp group toCE2 peer-as 3
set routing-instances purple protocols bgp group toCE2 neighbor 10.1.7.26
set routing-instances purple protocols bgp group toCE3 type external
set routing-instances purple protocols bgp group toCE3 peer-as 4
set routing-instances purple protocols bgp group toCE3 neighbor 10.1.8.30
set routing-instances purple routing-options multipath vpn-unequal-cost
    equal-external-internal
set policy-options policy-statement lb then load-balance per-packet
```

#### Device P

```
set interfaces ge-2/1/10 unit 14 family inet address 10.1.4.14/30
set interfaces ge-2/1/10 unit 14 family mpls
set interfaces ge-2/1/10 unit 14 description toPE2
set interfaces ge-2/1/10 unit 17 family inet address 10.1.5.17/30
set interfaces ge-2/1/10 unit 17 family mpls
set interfaces ge-2/1/10 unit 17 description toPE3
set interfaces lo0 unit 8 family inet address 1.1.1.5/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.8 passive
set protocols ospf area 0.0.0.0 interface ge-2/1/10.14 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/1/10.17 metric 5
set routing-options router-id 1.1.1.5
set routing-options autonomous-system 2
```

#### Device CE2

```
set interfaces ge-2/1/10 unit 22 family inet address 10.1.6.22/30
set interfaces ge-2/1/10 unit 22 family mpls
set interfaces ge-2/1/10 unit 22 description toPE2
set interfaces ge-2/1/10 unit 26 family inet address 10.1.7.26/30
set interfaces ge-2/1/10 unit 26 family mpls
set interfaces ge-2/1/10 unit 26 description toPE3
set interfaces lo0 unit 6 family inet address 1.1.1.6/32
set routing-options router-id 1.1.1.6
set routing-options autonomous-system 3
set protocols bgp group toAS2 type internal
set protocols bgp group toAS2 export send-direct
set protocols bgp group toAS2 peer-as 2
set protocols bgp group toAS2 neighbor 10.1.6.21
set protocols bgp group toAS2 neighbor 10.1.7.25
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

**Device CE3**

```

set interfaces ge-2/1/10 unit 30 family inet address 10.1.8.30/30
set interfaces ge-2/1/10 unit 30 family mpls
set interfaces ge-2/1/10 unit 30 description toPE3
set interfaces lo0 unit 7 family inet address 1.1.1.7/32
set routing-options router-id 1.1.1.7
set routing-options autonomous-system 4
set protocols bgp group toPE3 type internal
set protocols bgp group toPE3 export send-direct
set protocols bgp group toPE3 peer-as 2
set protocols bgp group toPE3 neighbor 10.1.8.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

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

To configure unequal-cost load balancing across the VPN setup:

1. Configure the router ID on Device CE1, and assign the device to its autonomous system.
 

```

[edit routing-options]
user@CE1# set routing-options router-id 1.1.1.1
user@CE1# set routing-options autonomous-system 1

```

Similarly, configure all other devices.
2. Configure BGP groups for traffic through the entire network.
  - a. Configure the BGP group for traffic to and from the MPLS network (CE devices).
 

```

[edit protocols bgp group toPE1]
user@CE1# set type external
user@CE1# set peer-as 2
user@CE1# set neighbor 10.1.1.2

```
  - b. Configure similar BGP groups (**toAS2** and **toPE3**) on Devices CE2 and CE3 by modifying the **peer-as** and **neighbor** statements accordingly.
  - c. Configure the BGP group for traffic through the MPLS network (PE devices).
 

```

[edit protocols bgp group toInternal]
user@PE1# set type internal
user@PE1# set family inet-vpn unicast
user@PE1# set local-address 1.1.1.2
user@PE1# set neighbor 1.1.1.3
user@PE1# set neighbor 1.1.1.4

```
  - d. Configure the same BGP group (**toInternal**) on Devices PE2 and PE3 by modifying the **local-address** and **neighbor** statements accordingly.
3. Configure a routing policy for exporting routes to and from the MPLS network (**send-direct** policy) and a policy for load balancing traffic network across the MPLS network (**lb** policy).

- a. Configure a policy (**send-direct**) for exporting routes from the routing table into BGP on Device CE1.

```
[edit policy-options policy-statement send-direct]
user@CE1# set from protocol direct
user@CE1# set then accept

[edit protocols bgp group toPE1]
user@CE1# set export send-direct
```

Similarly, configure the **send-direct** policy on Devices CE2 and CE3.

- b. Configure a policy (**lb**) for exporting routes from the routing table into the forwarding table on Device PE1.

The **lb** policy configures per-packet load balancing, which ensures that all next-hop addresses for a destination are installed in the forwarding table.

```
[edit policy-options policy-statement lb]
user@PE1# set then load-balance per-packet

[edit routing-options]
user@PE1# set forwarding-table export lb
```

Similarly, configure the **lb** policy on Devices PE2, and PE3.

4. Configure the following:
  - a. Configure the routing instance on the PE devices for exporting routes through the autonomous systems.
  - b. Include the **equal-external-internal** statement at the **[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]** hierarchy level to enable load balancing in the network.
  - c. Include the **vrf-table-label** statement at the **[edit routing-instances instance-name]** hierarchy level for filtering traffic prior to exiting the egress device (Device CE3).

#### Device PE1

```
[edit routing-instances purple]
user@PE1# set instance-type vrf
user@PE1# set interface ge-2/1/10.2
user@PE1# set route-distinguisher 2:1
user@PE1# set vrf-target target:2:1
user@PE1# set vrf-table-label
user@PE1# set protocols bgp group toCE1 type external
user@PE1# set protocols bgp group toCE1 peer-as 1
user@PE1# set protocols bgp group toCE1 neighbor 10.1.1.1
user@PE1# set routing-options multipath vpn-unequal-cost equal-external-internal
```

#### Device PE2

```
[edit routing-instances purple]
user@PE2# set instance-type vrf
user@PE2# set interface ge-2/1/10.21
user@PE2# set route-distinguisher 2:1
user@PE2# set vrf-target target:2:1
user@PE2# set vrf-table-label
```

```

user@PE2# set protocols bgp group toCE2 type external
user@PE2# set protocols bgp group toCE2 peer-as 3
user@PE2# set protocols bgp group toCE2 neighbor 10.1.6.22
user@PE2# set routing-options multipath vpn-unequal-cost equal-external-internal

```

### Device PE3

```

[edit routing-instances purple]
user@PE3# set instance-type vrf
user@PE3# set interface ge-2/1/10.25
user@PE3# set interface ge-2/1/10.29
user@PE3# set route-distinguisher 2:1
user@PE3# set vrf-target target:2:1
user@PE3# set vrf-table-label
user@PE3# set protocols bgp group toCE2 type external
user@PE3# set protocols bgp group toCE2 peer-as 3
user@PE3# set protocols bgp group toCE2 neighbor 10.1.7.26
user@PE3# set protocols bgp group toCE3 type external
user@PE3# set protocols bgp group toCE3 peer-as 4
user@PE3# set protocols bgp group toCE3 neighbor 10.1.8.30
user@PE3# set routing-options multipath vpn-unequal-cost equal-external-internal

```

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

```

user@PE3# show interfaces
ge-2/1/10 {
  unit 10 {
    description toPE1;
    family inet {
      address 10.1.3.10/30;
    }
    family mpls
  }
  unit 18 {
    description toP;
    family inet {
      address 10.1.5.18/30;
    }
    family mpls
  }
  unit 25 {
    description toCE2;
    family inet {
      address 10.1.7.25/30;
    }
    family mpls
  }
  unit 29 {
    description toCE3;
    family inet {
      address 10.1.8.29/30;
    }
    family mpls
  }
}

```

```
    }
  }
  lo0 {
    unit 7 {
      family inet {
        address 1.1.1.4/32;
      }
    }
  }
}

user@PE3# show protocols
mpls {
  interface all;
}
bgp {
  group toInternal {
    type internal;
    local-address 1.1.1.4;
    family inet {
      unicast;
    }
    family inet-vpn {
      unicast;
    }
    family route-target;
    neighbor 1.1.1.2;
    neighbor 1.1.1.3;
  }
}
ospf {
  area 0.0.0.0 {
    interface lo0.7 {
      passive;
    }
    interface ge-2/1/10.10 {
      metric 10;
    }
    interface ge-2/1/10.18 {
      metric 5;
    }
  }
}
ldp {
  interface all;
}

user@PE3# show policy-options
policy-statement lb {
  then {
    load-balance per-packet;
  }
}

user@PE3# show routing-instances
purple {
  instance-type vrf;
  interface ge-2/1/10.25;
```

```

interface ge-2/1/10.29;
route-distinguisher 2:1;
vrf-target target:2:1;
vrf-table-label;
routing-options {
    multipath {
        vpn-unequal-cost equal-external-internal;
    }
}
protocols {
    bgp {
        group toCE2 {
            type external;
            peer-as 3;
            neighbor 10.1.7.26;
        }
        group toCE3 {
            type external;
            peer-as 4;
            neighbor 10.1.8.30;
        }
    }
}
}

user@PE3# show routing-options
router-id 1.1.1.4;
autonomous-system 2;
forwarding-table {
    export lb;
}

```

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

## Verification

Confirm that the configuration is working properly.

- [Verifying BGP on page 223](#)
- [Verifying Load Balancing on page 225](#)
- [Verifying Load Balancing While Using IP Header Filtering on page 226](#)

### Verifying BGP

**Purpose** Verify that BGP is working.

**Action** From operational mode, run the **show route protocol bgp** command.

```

user@PE3> show route protocol bgp

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

purple.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only

```

+ = Active Route, - = Last Active, \* = Both

```

1.1.1.1/32      *[BGP/170] 04:47:14, localpref 100, from 1.1.1.2
                 AS path: 1 I
                 > to 10.1.3.9 via ge-2/1/10.10, Push 16
1.1.1.6/32      @[BGP/170] 00:13:28, localpref 100
                 AS path: 3 I
                 > to 10.1.7.26 via ge-2/1/10.25
                 [BGP/170] 00:10:36, localpref 100, from 1.1.1.3
                 AS path: 3 I
                 > to 10.1.5.17 via ge-2/1/10.18, Push 16, Push 299776(top)
1.1.1.7/32      *[BGP/170] 00:10:56, localpref 100
                 AS path: 4 I
                 > to 10.1.8.30 via ge-2/1/10.29
10.1.1.0/30     *[BGP/170] 04:47:14, localpref 100, from 1.1.1.2
                 AS path: I
                 > to 10.1.3.9 via ge-2/1/10.10, Push 16
10.1.6.20/30    *[BGP/170] 04:47:03, localpref 100, from 1.1.1.3
                 AS path: I
                 > to 10.1.5.17 via ge-2/1/10.18, Push 16, Push 299776(top)
                 [BGP/170] 00:13:28, localpref 100
                 AS path: 3 I
                 > to 10.1.7.26 via ge-2/1/10.25
10.1.7.24/30    [BGP/170] 00:13:28, localpref 100
                 AS path: 3 I
                 > to 10.1.7.26 via ge-2/1/10.25
10.1.8.28/30    [BGP/170] 00:10:56, localpref 100
                 AS path: 4 I
                 > to 10.1.8.30 via ge-2/1/10.29

```

mpls.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

+ = Active Route, - = Last Active, \* = Both

```

2:1:1.1.1.1/32  *[BGP/170] 04:47:14, localpref 100, from 1.1.1.2
                 AS path: 1 I
                 > to 10.1.3.9 via ge-2/1/10.10, Push 16
2:1:1.1.1.6/32  *[BGP/170] 00:10:36, localpref 100, from 1.1.1.3
                 AS path: 3 I
                 > to 10.1.5.17 via ge-2/1/10.18, Push 16, Push 299776(top)
2:1:10.1.1.0/30 *[BGP/170] 04:47:14, localpref 100, from 1.1.1.2
                 AS path: I
                 > to 10.1.3.9 via ge-2/1/10.10, Push 16
2:1:10.1.6.20/30 *[BGP/170] 04:47:03, localpref 100, from 1.1.1.3

```

The output lists the BGP routes installed into the routing table. The lines of output that start with **1.1.1.1/32**, **10.1.1.0/30**, and **2:1:1.1.1/32** show the BGP routes to Device CE1, which is in AS1. The lines of output that start with **1.1.1.6/32**, **2:1:1.1.6/32**, and **2:1:10.1.6.20/30** show the BGP routes to Device CE2, which is in AS3. The line of output that starts with **1.1.1.7/32** shows the BGP route to Device CE3, which is in AS4.

**Meaning** BGP is functional in the network.

**Verifying Load Balancing**

**Purpose** Verify that forwarding is taking place in both directions by checking:

- If both next hops are installed in the forwarding table for a route.
- If external BGP routes are installed in the forwarding table for a route.

**Action** From operational mode, run the **show route forwarding-table** and **show route forwarding-table destination <destination IP>** commands.

```
user@PE3> show route forwarding-table
```

```
Router: PE3
Routing table: default.inet
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
default          perm  0                rjct  593   1
0.0.0.0/32       perm  0                dscd  579   1
1.1.1.2/32       user  1 10.1.3.9          ucst  999   8 ge-2/1/10.10
1.1.1.3/32       user  1 10.1.5.17         ucst  1243  12 ge-2/1/10.18
1.1.1.4/32       intf  0 1.1.1.4           locl  895   1
1.1.1.5/32       user  1 10.1.5.17         ucst  1243  12 ge-2/1/10.18
10.1.2.4/30      user  0                ulst  1048580 2
                  10.1.3.9          ucst  999   8 ge-2/1/10.10
                  10.1.5.17         ucst  1243  12 ge-2/1/10.18
10.1.3.8/30      intf  0                rslv  899   1 ge-2/1/10.10
10.1.3.8/32      dest  0 10.1.3.8          recv  897   1 ge-2/1/10.10
10.1.3.9/32      dest  0 0.6.80.3.0.21.59.d.c5.d9.0.21.59.d.c5.da.8.0 ucst  999   8 ge-2/1/10.10
10.1.3.10/32     intf  0 10.1.3.10         locl  898   2
10.1.3.10/32     dest  0 10.1.3.10         locl  898   2
10.1.3.11/32     dest  0 10.1.3.11         bcst  896   1 ge-2/1/10.10
10.1.4.12/30     user  0 10.1.5.17         ucst  1243  12 ge-2/1/10.18
10.1.5.16/30     intf  0                rslv  903   1 ge-2/1/10.18
10.1.5.16/32     dest  0 10.1.5.16         recv  901   1 ge-2/1/10.18
10.1.5.17/32     dest  0 0.e.80.3.0.21.59.d.c5.d9.0.21.59.d.c5.da.8.0 ucst  1243  12 ge-2/1/10.18
10.1.5.18/32     intf  0 10.1.5.18         locl  902   2
10.1.5.18/32     dest  0 10.1.5.18         locl  902   2
10.1.5.19/32     dest  0 10.1.5.19         bcst  900   1 ge-2/1/10.18
224.0.0.0/4      perm  2                mdsc  592   1
224.0.0.1/32     perm  0 224.0.0.1         mcst  576   3
224.0.0.5/32     user  1 224.0.0.5         mcst  576   3
255.255.255.255/32 perm  0                bcst  577   1
```

```
Router: PE3
Routing table: __master.anon__.inet
```

```
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
default          perm  0                rjct  909   1
0.0.0.0/32       perm  0                dscd  907   1
224.0.0.0/4      perm  0                mdsc  908   1
224.0.0.1/32     perm  0 224.0.0.1         mcst  904   1
255.255.255.255/32 perm  0                bcst  905   1
```

```
Router: PE3
Routing table: purple.inet
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
```

```

default          perm    0          rjct   918    1
0.0.0.0/32       perm    0          dscd   916    1
1.1.1.1/32       user    0          indr  1048576   3
                  10.1.3.9    Push 16 1187   2 ge-2/1/10.10
1.1.1.6/32       user    0          ulst  1048587   2
                  10.1.7.26    ucst  1239    4 ge-2/1/10.25
                  10.1.5.17    indr  1048579   3
                  Push 16, Push 299776(top) 1306
                2 ge-2/1/10.18
1.1.1.7/32       user    0 10.1.8.30    ucst  1299    4 ge-2/1/10.29
299808(S=0)      user    0 10.1.5.17    Pop   1304    2 ge-2/1/10.18
...

```

In the **default.inet** routing table, which is the forwarding table, the line of output that starts with **10.1.2.4/30** shows that for a route to Device PE2 in the same AS, two next hops are installed in the table: **10.1.3.9** and **10.1.5.17**.

In the **purple.inet** routing table, which is the external routing table, the line of output that starts with **1.1.1.6/32** shows that for a route to Device CE2 in AS3, an internal next hop of **10.1.5.17** and an external next hop of **10.1.7.26** are installed in the table. This indicates that both internal and external BGP routes are operational in the network.

```
user@PE3> show route forwarding-table destination 10.1.2.6
```

```

Router: PE3
Routing table: default.inet
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
10.1.2.4/30      user    0          10.1.3.9          ulst 1048580   2
                  10.1.5.17          ucst   999    8 ge-2/1/10.10
                  ucst  1243   12 ge-2/1/10.18

```

```

Router: PE3
Routing table: __master.anon__.inet
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
default          perm    0          rjct   909    1

```

```

Router: PE3
Routing table: purple.inet
Internet:
Destination      Type RtRef Next hop          Type Index NhRef Netif
default          perm    0          rjct   918    1

```

The line of output that starts with **10.1.2.4/30** shows that for a route from Device PE3 to Device PE2 in the same AS, two next hops are installed in the table: **10.1.3.9** through the **ge-2/1/10.10** interface, and **10.1.5.17** through the **ge-2/1/10.18** interface.

**Meaning** Multiple next hops for a route, including external BGP routes, are installed in the forwarding tables.

### *Verifying Load Balancing While Using IP Header Filtering*

**Purpose** Verify that filtered traffic reaches the egress CE devices after load balancing has been configured on the PE devices.

**Action** Configure a firewall filter on Device PE3 on the interface connecting to Device CE2.

```
[edit firewall family inet filter filterPE3 term a]
user@PE3# set from protocol tcp
user@PE3# set from source-port-except bgp
user@PE3# set from destination-port-except bgp
user@PE3# set then count filterPE3
user@PE3# set then accept

[edit firewall family inet filter filterPE3 term b]
user@PE3# set then accept

[edit interfaces ge-2/1/10 unit 25]
user@PE3# set family inet filter output filterPE3
```

Similarly, configure a firewall filter on Device PE3 on the interface facing Device CE3, and another on Device PE2 on the interface facing Device CE2.

Count the packets exiting the egress interfaces on Devices PE2 and PE3 by using the **show firewall filter <filter name> counter <counter name>** operational mode command. The output confirms if load balancing takes place with IP header filtering configured (enabled by the **vrf-table-label** statement). If all transmitted packets have been load-balanced between the paths PE3->CE2, PE3->CE3, and PE2->CE2, then it means that the IP header filtering feature works in a load-balanced Layer 3 network.

You can clear the counter by using the **clear firewall filter <filter name> counter <counter name>** operational mode command.

**Meaning** Load balancing takes place with IP header filtering configured.

## Example: Disabling Normal TTL Decrementing in a VRF Routing Instance

This example shows how to disable TTL decrementing in a single VRF routing instance in a Layer 3 VPN scenario.

- [Requirements on page 227](#)
- [Overview on page 227](#)
- [Configuration on page 229](#)
- [Verification on page 234](#)

### Requirements

Before you begin:

- Configure the router interfaces. See the *Network Interfaces Configuration Guide*.

### Overview

To diagnose networking problems related to VPNs, it can be useful to disable normal time-to-live (TTL) decrementing. The IP header includes a TTL field that serves as a hop counter. At every routed hop, the TTL is decremented by one; if the TTL reaches zero before the packet reaches its destination, the packet is discarded and (optionally) an ICMP TTL exceeded message is sent to the source. MPLS labels also have a TTL field. MPLS routers copy the TTL of an IP packet when it enters a label-switched path (LSP).

An IP packet with a TTL of 27 receives an MPLS label with a TTL of 27. Junos OS decrements the MPLS TTL of an MPLS-encapsulated packet in place of the IP TTL, at every label-switched hop. Because the MPLS TTL is copied (or propagated) from the IP TTL, a traceroute lists every hop in the path, be it routed or label-switched. When the packet exits the LSP, the decremented MPLS TTL is propagated back into the IP TTL field.

By default, TTL propagation is enabled. The global **no-propagate-ttl** statement disables TTL propagation at the router level and affects all RSVP-signalled or LDP-signalled LSPs. When a router acts as an ingress router for an LSP and the router configuration includes the **no-propagate-ttl** statement, the router pushes an MPLS header with a TTL value of 255, regardless of the IP packet TTL. When a router acts as the penultimate router, it pops the MPLS header without propagating the MPLS TTL into the IP packet. Thus the IP packet TTL value is preserved, regardless of the hop count of the LSP.

Instead of configuring TTL propagation behavior at the router level, you can configure the behavior for the routes in a VRF routing instance. This example shows how to disable TTL propagation for the routes in a single VRF routing instance instead of at the global router level.

The per-VRF configuration takes precedence over the global router configuration. If you disable TTL propagation on the router and explicitly enable TTL propagation for a single VRF routing instance, TTL propagation is in effect for that routing instance. To explicitly enable TTL propagation on a VRF routing instance, include the **vrf-propagate-ttl** statement in the routing instance.

When you change the TTL propagation behavior, old next hops for VRF routes are deleted from the inet.3 routing table and new next hops are added.

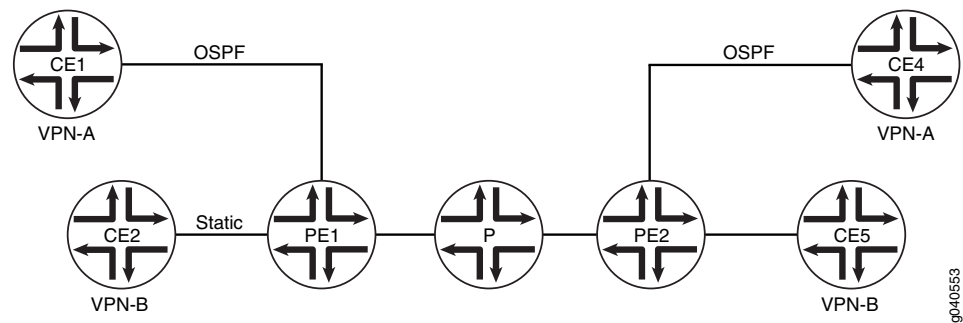
You need only configure the **vrf-propagate-ttl** or **no-vrf-propagate-ttl** statement on the ingress routers.

---

### Topology Diagram

Figure 33 on page 229 shows the topology used in this example. Router PE1 and Router PE2 have two VPNs---VPN-A and VPN-B. Devices CE1 and CE4 belong to VPN-A. Devices CE2 and CE5 belong to VPN-B. In this example, Router PE1 has TTL propagation disabled on VPN-A but not on VPN-B. Packets received by PE1 on the interface connected to CE1 have TTL propagation disabled. This example shows the configuration on Router PE1. You do not need to include the **no-vrf-propagate-ttl** statement on the egress router (PE2).

Figure 33: Disabling TTL Propagation for a Single VPN



## Configuration

### CLI Quick Configuration

To quickly disable TTL propagation in a VRF routing instance, copy the following commands and paste the commands into the CLI.

```
[edit]
set interfaces lo0 unit 0 family inet address 10.255.179.45/32 primary
set protocols mpls interface all
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.179.45
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 10.255.179.71
set protocols ospf area 0.0.0.0 interface fe-1/1/2.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ldp interface all
set policy-options policy-statement VPN-A-export term a from protocol ospf
set policy-options policy-statement VPN-A-export term a from interface ge-1/2/0.0
set policy-options policy-statement VPN-A-export term a then community add VPN-A
set policy-options policy-statement VPN-A-export term a then accept
set policy-options policy-statement VPN-A-export term b then reject
set policy-options policy-statement VPN-A-import term a from protocol bgp
set policy-options policy-statement VPN-A-import term a from community VPN-A
set policy-options policy-statement VPN-A-import term a then accept
set policy-options policy-statement VPN-A-import term b then reject
set policy-options policy-statement VPN-B-export term a from protocol static
set policy-options policy-statement VPN-B-export term a then community add VPN-B
set policy-options policy-statement VPN-B-export term a then accept
set policy-options policy-statement VPN-B-export term b then reject
set policy-options policy-statement VPN-B-import term a from protocol bgp
set policy-options policy-statement VPN-B-import term a from community VPN-B
set policy-options policy-statement VPN-B-import term a then accept
set policy-options policy-statement VPN-B-import term b then reject
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set policy-options community VPN-A members target:1:100
set policy-options community VPN-B members target:1:200
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A route-distinguisher 10.255.179.45:100
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A no-vrf-propagate-ttl
```

```

set routing-instances VPN-A vrf-import VPN-A-import
set routing-instances VPN-A vrf-export VPN-A-export
set routing-instances VPN-A protocols ospf export bgp-to-ospf
set routing-instances VPN-A protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set routing-instances VPN-B instance-type vrf
set routing-instances VPN-B interface so-0/1/0.0
set routing-instances VPN-B route-distinguisher 10.255.179.45:300
set routing-instances VPN-B vrf-import VPN-B-import
set routing-instances VPN-B vrf-export VPN-B-export
set routing-instances VPN-B routing-options static route 10.255.179.15/32 next-hop
    so-0/1/0.0
set routing-options autonomous-system 1

```

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

To configure a flow map:

1. Configure the loopback interface.

```

[edit]
user@PE1# edit interfaces
[edit interfaces]
user@PE1# set lo0 unit 0 family inet address 10.255.179.45/32 primary
user@PE1# exit

```

2. Configure the routing protocols.

The internal BGP neighbor address is the loopback interface address of Router PE2 in [Figure 33 on page 229](#).

```

[edit]
user@PE1# edit protocols
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 10.255.179.45
user@PE1# set bgp group ibgp family inet-vpn unicast
user@PE1# set bgp group ibgp neighbor 10.255.179.71
user@PE1# set ospf area 0.0.0.0 interface fe-1/1/2.0
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE1# set ospf area 0.0.0.0 interface lo0.0
user@PE1# set ldp interface all
user@PE1# exit

```

3. Configure routing policies for VPN-A and VPN-B.

```

[edit]
user@PE1# edit policy-options
[edit policy-options]
user@PE1# set policy-statement VPN-A-export term a from protocol ospf
user@PE1# set policy-statement VPN-A-export term a from interface ge-1/2/0.0
user@PE1# set policy-statement VPN-A-export term a then community add VPN-A
user@PE1# set policy-statement VPN-A-export term a then accept
user@PE1# set policy-statement VPN-A-export term b then reject
user@PE1# set policy-statement VPN-A-import term a from protocol bgp

```

```

user@PE1# set policy-statement VPN-A-import term a from community VPN-A
user@PE1# set policy-statement VPN-A-import term a then accept
user@PE1# set policy-statement VPN-A-import term b then reject
user@PE1# set policy-statement VPN-B-export term a from protocol static
user@PE1# set policy-statement VPN-B-export term a then community add VPN-B
user@PE1# set policy-statement VPN-B-export term a then accept
user@PE1# set policy-statement VPN-B-export term b then reject
user@PE1# set policy-statement VPN-B-import term a from protocol bgp
user@PE1# set policy-statement VPN-B-import term a from community VPN-B
user@PE1# set policy-statement VPN-B-import term a then accept
user@PE1# set policy-statement VPN-B-import term b then reject
user@PE1# set policy-statement bgp-to-ospf from protocol bgp
user@PE1# set policy-statement bgp-to-ospf then accept
user@PE1# set community VPN-A members target:1:100
user@PE1# set community VPN-B members target:1:200
user@PE1# exit

```

4. Configure the VPN-A and VPN-B routing instances, including the **no-vrf-propagate-ttl** statement in VPN-A.

```

[edit]
user@PE1# edit routing-instances
[edit routing-instances]
user@PE1# set VPN-A instance-type vrf
user@PE1# set VPN-A interface ge-1/2/0.0
user@PE1# set VPN-A route-distinguisher 10.255.179.45:100
user@PE1# set VPN-A interface ge-1/2/0.0
user@PE1# set VPN-A no-vrf-propagate-ttl
user@PE1# set VPN-A vrf-import VPN-A-import
user@PE1# set VPN-A vrf-export VPN-A-export
user@PE1# set VPN-A protocols ospf export bgp-to-ospf
user@PE1# set VPN-A protocols ospf area 0.0.0.0 interface ge-1/2/0.0
user@PE1# set VPN-B instance-type vrf
user@PE1# set VPN-B interface so-0/1/0.0
user@PE1# set VPN-B route-distinguisher 10.255.179.45:300
user@PE1# set VPN-B vrf-import VPN-B-import
user@PE1# set VPN-B vrf-export VPN-B-export
user@PE1# set VPN-B routing-options static route 10.255.179.15/32 next-hop
so-0/1/0.0
user@PE1# exit

```

5. Define the local autonomous system.

```

[edit]
user@PE1# edit routing-options
[edit routing-options]
user@PE1# set autonomous-system 1
user@PE1# exit

```

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

```

[edit]
user@PE1# commit

```

## Results

---

Confirm your configuration by entering the **show interfaces**, **show policy-options**, **show protocols**, **show routing-instances**, and **show routing-options** commands.

```
user@PE1# show interfaces
lo0 {
  unit 0 {
    family inet {
      address 10.255.179.45/32 {
        primary;
      }
    }
  }
}

user@PE1# show policy-options
policy-statement VPN-A-export {
  term a {
    from {
      protocol ospf;
      interface ge-1/2/0.0;
    }
    then {
      community add VPN-A;
      accept;
    }
  }
  term b {
    then reject;
  }
}
policy-statement VPN-A-import {
  term a {
    from {
      protocol bgp;
      community VPN-A;
    }
    then accept;
  }
  term b {
    then reject;
  }
}
policy-statement VPN-B-export {
  term a {
    from protocol static;
    then {
      community add VPN-B;
      accept;
    }
  }
  term b {
    then reject;
  }
}
```

```

}
policy-statement VPN-B-import {
  term a {
    from {
      protocol bgp;
      community VPN-B;
    }
    then accept;
  }
  term b {
    then reject;
  }
}
policy-statement bgp-to-ospf {
  from protocol bgp;
  then accept;
}
community VPN-A members target:1:100;
community VPN-B members target:1:200;

user@PE1# show protocols
mpls {
  interface all;
}
bgp {
  group ibgp {
    type internal;
    local-address 10.255.179.45;
    family inet-vpn {
      unicast;
    }
    neighbor 10.255.179.71;
  }
}
ospf {
  area 0.0.0.0 {
    interface fe-1/1/2.0;
    interface fxp0.0 {
      disable;
    }
    interface lo0.0;
  }
}
ldp {
  interface all;
}

user@PE1# show routing-instances
VPN-A {
  instance-type vrf;
  interface ge-1/2/0.0;
  no-vrf-propagate-ttl;
  route-distinguisher 10.255.179.45:100;
  vrf-import VPN-A-import;
  vrf-export VPN-A-export;
  protocols {
    ospf {

```

```
        export bgp-to-ospf;
        area 0.0.0.0 {
            interface ge-1/2/0.0;
        }
    }
}
VPN-B {
    instance-type vrf;
    interface so-0/1/0.0;
    route-distinguisher 10.255.179.45:300;
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
    routing-options {
        static {
            route 10.255.179.15/32 next-hop so-0/1/0.0;
        }
    }
}

user@PE1# show routing-options
autonomous-system 1;
```

## Verification

To verify the operation, run the following commands:

- See the **TTL Action** field in the output of the **show route extensive table VPN-A** command.
- See the **TTL Action** field in the output of the **show route extensive table VPN-B** command.
- On Device CE1, run the **traceroute** command to Device CE4's loopback address.
- On Device CE4, run the **traceroute** command to Device CE1's loopback address.

**Related Documentation**

- *Disabling Normal TTL Decrementing*

---

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

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

- [Requirements on page 234](#)
- [Overview on page 235](#)
- [Configuration on page 236](#)
- [Verification on page 237](#)

## Requirements

This example uses Junos OS Release 10.0 or later.

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol. See the *Junos OS Routing Protocols Library for Routing Devices*.
- Configure a BGP session for multiple route types. For example, configure the session for both family **inet** routes and family **inet-vpn** routes. See *Configuring IBGP Sessions Between PE Routers in VPNs* and [“Configuring Layer 3 VPNs to Carry IPv6 Traffic” on page 46](#).

## Overview

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

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

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

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

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

```
from {
  route-filter 0.0.0.0/0 exact;
}
then {
  next-hop self;
  accept;
}
```

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

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

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

## Configuration

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

**inet Example**

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

**Inet-vpn Example**

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

**inet6 Example**

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

**Inet6-vpn Example**

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

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

To configure a flow map:

1. Configure the family qualifier.

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

2. Configure the route filter.

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

3. Configure the policy actions.

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

4. Apply the policy.

```
[edit protocols bgp]
```

```
user@host# set import specific-family
```

## Results

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

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

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

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

## Verification

To verify the configuration, run the following commands:

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

### Related Documentation

- *Understanding Route Filters for Use in Routing Policy Match Conditions*
- *Route Filter Match Conditions*
- *Example: Configuring Policy Chains and Route Filters*
- *Example: Configuring the MED Using Route Filters*
- *Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF*

## Example: Configuring Route Resolution on PE Routers

This example shows how to configure a routing table to accept routes from specific routing tables. It also shows how to configure a routing table to use specific import policies to produce a route resolution table to resolve routes.

- [Requirements on page 238](#)
- [Overview on page 238](#)
- [Configuration on page 238](#)
- [Verification on page 239](#)

### Requirements

Before you begin, configure a Layer 3 VPN, as shown in one of the following examples:

- [Example: Configuring Interprovider Layer 3 VPN Option A on page 405](#)
- [Example: Configuring Interprovider Layer 3 VPN Option B on page 425](#)

### Overview

One method to achieve IPv4 route scaling is to modify how BGP routes are added to the forwarding tables. By default, the Routing Protocol Process (rpd) adds all the routes in inet.0 and inet.3 to the resolution tree. Normally, this includes the resolved IPv4 BGP routes, which can increase memory consumption. To achieve better scaling for IPv4 routes, this example shows how to configure the Junos OS so that resolved BGP routes are not added to the resolution tree. This is achieved by applying an import policy on the route resolution table, which ensures it does not accept any BGP routes from inet.0.

You would apply this configuration to all provider edge (PE) routers in the Layer 3 VPN.

### Configuration

<b>CLI Quick Configuration</b>	To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the <b>[edit]</b> hierarchy level.
<b>PE Router</b>	<pre>set policy-options policy-statement protocol-bgp from protocol bgp set policy-options policy-statement protocol-bgp then reject set routing-options resolution rib inet.0 import protocol-bgp</pre>
<b>Step-by-Step Procedure</b>	<p>The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see <i>Using the CLI Editor in Configuration Mode</i> in the <i>CLI User Guide</i>.</p> <p>To configure route resolution:</p> <ol style="list-style-type: none"> <li>1. Configure the routing policy. <pre>[edit policy-options policy-statement protocol-bgp] user@PE# set from protocol bgp</pre> </li> </ol>

```
user@PE# set then reject
```

2. Apply the routing policy.

```
[edit routing-options resolution]
user@PE# set rib inet.0 import protocol-bgp
```

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

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

## Results

Confirm your configuration by issuing the **show policy-options** and **show routing-options** commands.

```
user@PE# show policy-options
policy-statement protocol-bgp {
  from protocol bgp;
  then reject;
}

user@PE# show routing-options
resolution {
  rib inet.0 {
    import protocol-bgp;
  }
}
```

## Verification

Confirm that the configuration is working properly by running the following commands:

- *show route*
- *show route forwarding-table*
- *show route resolution*

### Related Documentation

- [Example: Configuring Route Resolution on Route Reflectors on page 239](#)

## Example: Configuring Route Resolution on Route Reflectors

This example shows how to change the default resolution behavior on a route reflector (RR) to use inet.0 for next-hop resolution instead of inet.3.

- [Requirements on page 240](#)
- [Overview on page 240](#)
- [Configuration on page 241](#)
- [Verification on page 242](#)

## Requirements

Before you begin, configure a Layer 3 VPN, as shown in one of the following examples:

- [Example: Configuring Interprovider Layer 3 VPN Option A on page 405](#)
- [Example: Configuring Interprovider Layer 3 VPN Option B on page 425](#)

## Overview

One scenario for route resolution is when you have a label-switched path configured from an RR to a provider edge (PE) router, or when the PE routers only peer with the RR. This can result in routes being hidden. To resolve this issue, you can change the default resolution behavior to use `inet.0` for next-hop resolution.

By default, the `bgp.l3vpn.0` routing table stores all VPN-IPv4 unicast routes. This table is present on any router that has Layer 3 VPNs configured, including PE routers and RRs.

When a Layer 3 VPN router receives a route from another Layer 3 VPN router, it places the route into its `bgp.l3vpn.0` routing table. The route is resolved using the information in the `inet.3` routing table. This means that when BGP receives a route destined for table `bgp.l3vpn.0`, the protocol nexthop (received BGP nexthop) has its forwarding nexthop recursively determined from the `inet.3` table. The resulting route is converted into IPv4 format and redistributed to all *routing-instance-name.inet.0* routing tables if it matches the VRF import policy.

On an RR with no attached customer edge (CE) routers, the **`resolution rib bgp.l3vpn.0 resolution-ribs inet.0`** configuration causes routes in `bgp.l3vpn.0` to use the information in `inet.0` instead of `inet.3` to resolve routes. You should not use this configuration on a router that is directly attached to a CE router. In other words, do not use **`resolution rib bgp.l3vpn.0 resolution-ribs inet.0`** on a PE router.

If you want both `inet.0` and `inet.3` to be used, you must configure both, as in **`set resolution rib bgp.l3vpn.0 resolution-ribs [inet.0 inet.3]`**.

In this example, the policy **`POLICY-limit-resolve-routes`** limits the route resolution to only routes learned through IS-IS. If you omit the import policy, all routes in `inet.0` are evaluated and potentially used to resolve the protocol next hop. If you do not want to resolve against all entries, you use a policy to filter for a subset of the routes from the tables that are used for route resolution.

One common example is when you resolve against all routes in `inet.0`, except the default route (0/0).

Although the **`import`** statement is used in this configuration, no routes are imported or copied. Rather, the **`import policy-name`** configuration limits the set of possible routes that can be considered for route resolution.

The **`resolution rib bgp.l3vpn.0 resolution-ribs inet.0`** configuration is useful when a BGP RR is not in the forwarding path. In other words, there are no ingress LSPs at the RR. Consider the case where RSVP is the label signaling protocol, and RSVP is configured full mesh at the edge routers. The RR needs to be able to reflect the routes. To do so,

BGP is expected to perform a route resolvability check. If a Layer 3 VPN route is received but the nexthop is not in the inet.3 table, the route cannot be resolved. Because the router is not in the forwarding path, an effective workaround is to use the information in inet.0. The metric information in inet.0 is useful for choosing the best route, even though it cannot be used for forwarding.

An alternative approach is to make sure that LSPs are provisioned to the RR. This happens automatically if you configure LDP.

## Configuration

### CLI Quick Configuration

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

### Route Reflector

```
set routing-options resolution rib bgp.l3vpn.0 resolution-ribs inet.0
set routing-options resolution rib inet.0 import POLICY-limit-resolve-routes
set policy-options policy-statement POLICY-limit-resolve-routes term isis from protocol isis
set policy-options policy-statement POLICY-limit-resolve-routes term isis then accept
set policy-options policy-statement POLICY-limit-resolve-routes then reject
```

### Step-by-Step Procedure

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

To configure route resolution:

1. Configure bgp.l3vpn.0 to use the information in inet.0 instead of inet.3 to resolve routes.

```
[edit routing-options resolution rib bgp.l3vpn.0]
user@RR# set resolution-ribs inet.0
```

2. (Optional) Configure the routing policy.

```
[edit policy-options policy-statement POLICY-limit-resolve-routes]
user@RR# set term isis from protocol isis
user@RR# set term isis then accept
user@RR# set then reject
```

3. (Optional) Apply the policy.

```
[edit routing-options resolution rib inet.0]
user@RR# set import POLICY-limit-resolve-routes
```

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

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

## Results

Confirm your configuration by issuing the **show policy-options** and **show routing-options** commands.

```
user@RR# show policy-options
policy-statement POLICY-limit-resolve-routes {
  term isis {
    from protocol isis;
    then accept;
  }
  then reject;
}

user@RR# show routing-options
resolution {
  rib bgp.l3vpn.0 {
    resolution-ribs inet.0;
  }
  rib inet.0 {
    import POLICY-limit-resolve-routes;
  }
}
```

## Verification

Confirm that the configuration is working properly by running the following commands:

- *show route*
- *show route forwarding-table*
- *show route resolution*

### Related Documentation

- [Example: Configuring BGP Route Reflectors](#)
- [Example: Configuring Route Resolution on PE Routers on page 238](#)

---

## Example: Configuring Link Protection with Host Fast Reroute

- [Understanding Host Fast Reroute on page 242](#)
- [Example: Configuring Link Protection with Host Fast Reroute on page 247](#)

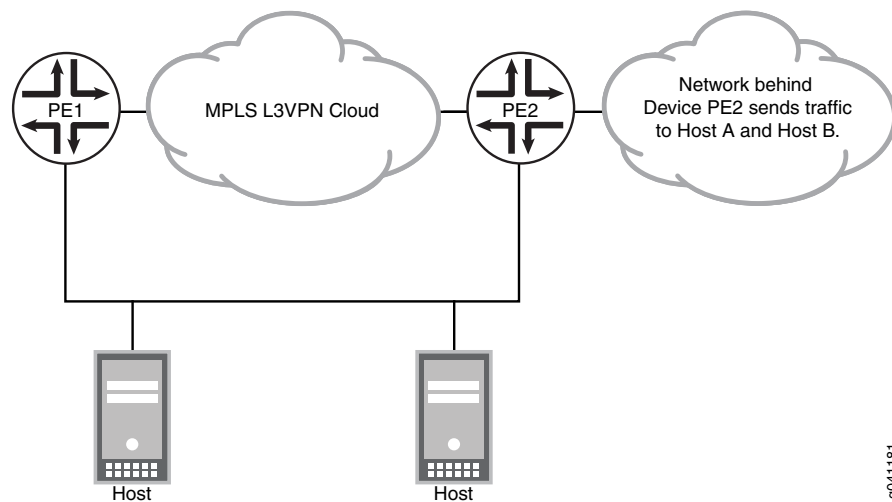
### Understanding Host Fast Reroute

Host fast reroute (HFRR) adds a precomputed protection path into the Packet Forwarding Engine (PFE), such that if a link between a provider edge device and a server farm becomes unusable for forwarding, the PFE can use another path without having to wait for the router or the protocols to provide updated forwarding information. This precomputed protection path is often called a repair or a backup path.

HFRR is a technology that protects IP endpoints on multipoint interfaces, such as Ethernet. This technology is important in datacenters where fast service restoration for server endpoints is critical. After an interface or a link goes down, HFRR enables the local repair time to be approximately 50 milliseconds.

Consider the network topology shown in [Figure 34 on page 243](#).

Figure 34: Host Fast Reroute



g041181

Routing devices create host route forwarding entries triggered by the Address Resolution Protocol (ARP) and IPv6 Neighbor Discovery Protocol (NDP). HFRR augments the host routes with backup next hops supplied by routing protocols. These backup next hops enable arriving traffic to keep flowing while the network reconverges.

Traffic flows from networks connected to the provider edge devices, PE1 and PE2, to host A and host B. This traffic is protected with HFRR. If the link goes down between device PE2 and the host servers, traffic is rerouted through device PE1 to the host servers. In the topology, host A and host B represent LAN PCs, collectively known as a server farm. The PE devices are routers with a Layer 3 VPN configured between them. Device PE1 learns about the directly connected hosts by way of ARP or the IPv6 NDP.

Device PE2 also has information about the server farm network and advertises this information to Device PE1. This advertisement is transmitted through the Layer 3 VPN using internal BGP (IBGP). On Devices PE1 and PE2, this route is considered a direct route to the server farm subnet.

Device PE1 uses the host routes learned through ARP and NDP to send traffic to the host machines in the server farm. If the link between Device PE1 and the server farm is disrupted and if HFRR is not configured, the routing device finds the next best route, which is the IBGP route. This implementation results in traffic loss for an interval until the update occurs and the network reconverges. HFRR configured on Device PE1 resolves this issue by augmenting the ARP and NDP routes with a backup path so that traffic can continue to be forwarded without interruption.

The backup path in this particular topology is the IBGP Layer 3 VPN route. In an actual deployment, Device PE2 can also configure link protection for its directly connected server farm network, and Device PE1 can advertise reachability to the server farm through itself using the Layer 3 VPN routes to Device PE2. Therefore, HFRR should be enabled on both Device PE1 and Device PE2. Also, Device PE1 and Device PE2 should both advertise reachability to the server farm through BGP.

A temporary routing loop can develop between the PE devices if, for example, the link between Device PE1 to the server farm and the link between Device PE2 to the server farm both go down at same time. The loop can continue until BGP on both ends learns that the server farm subnet is down and withdraws the BGP routes.

### ARP Prefix Limit and Blackout Supplementary Timeout

---

When you configure HFRR profiles, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore, FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices. The ARP prefix limit does not limit ARP routes in the forwarding table. It does, however, limit the number of ARP routes that Junos OS reads for a profile and therefore limits the number of HFRR routes that the routing process (rpd) creates in the routing table and the forwarding table.

The ARP prefix limit is applied to each HFRR profile. It does not limit the total count of all ARP/HFRR routes in the routing table. It only limits the number of ARP/HFRR routes for each HFRR profile.

There are two configuration statements ([global-arp-prefix-limit](#) and [arp-prefix-limit](#)) that set the ARP prefix limit, one at the global **[edit routing-options host-fast-reroute]** hierarchy level and the other at the **[edit routing-instances instance-name routing-options interface interface-name]** hierarchy level, respectively. The global **global-arp-prefix-limit** statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The **arp-prefix-limit** statement overrides the **global-arp-prefix-limit** for that HFRR profile for that protected interface.

When the number of ARP routes in an HFRR profile reaches 80% of the configured ARP prefix limit, a warning message is sent to the system log. The warning message is displayed for any subsequent ARP route added to that HFRR profile if the ARP prefixes remain at greater than 80% of the configured value.

When the number of ARP routes in an HFRR profile reaches 100% of the configured ARP prefix limit for an HFRR profile, another warning message is sent to the system log. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.

After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.

There are global and per-HFRR CLI statements ([global-supplementary-blackout-timer](#) and [supplementary-blackout-timer](#)). The global value is at the **[edit routing-options host-fast-reroute]** hierarchy level and applies to all HFRR profiles on the routing device. The supplementary blackout timer configured for the routing-instance interface at the **[edit routing-instances instance-name routing-options interface interface-name]** hierarchy level overrides the global value for that HFRR profile only.

When the blackout timer expires, the HFRR profile is reactivated, and Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.

If an HFRR profile is blacklisted and is in the deactivated state, a reevaluation of the ARP state is performed during every commit operation or whenever the routing process (rpd) is restarted with the **restart routing** command.

### Primary Route and Backup Route Candidates

The primary route for the HFRR next hop is provided by the ARP and IPv6 NDP routes. These are /32 or /128 routes. The backup route is an exact prefix match of the address configured on the local interface. For example, if the local address configured is 10.0.0.5/24, the routing device looks for an exact match of prefix 10.0.0.0 with a prefix length of 24 for selection of backup route.

Constraints for backup route selection are as follows:

- Must be a prefix matching the same subnet address configured on the routing device's HFRR-enabled interface.
- The remote end must not have route aggregation (also known as summarization) configured. For example, if the remote end combines two or more /24 subnets to advertise a subnet with a prefix length smaller than /24, the Junos OS does not select this summarized route to be a backup route.
- If there is another route in the routing table learned by another protocol with a longest-prefix match for the /32 or /128 (ARP or NDP) route, that route is not selected to be a backup candidate. For example, suppose that the local interface address is 10.0.0.5/24. Also suppose that the routing table contains an IBGP route with a prefix of 10.0.0.0/24 and an OSPF route with a prefix of 10.0.0.0/28. Even though the /28 route is a better route for certain prefixes in the subnet, the Junos OS does not consider 10.0.0.0/28 to be a backup candidate. The IBGP route becomes the backup candidate for all host routes. However, after the global repair, the OSPF route is used for forwarding.

In short, the backup candidate must be a route with the same prefix as the subnet local interface that you are protecting with HFRR.

### Backup Path Selection Policy

Only Layer 3 VPN routes are considered for backup selection. HFRR uses the usual BGP path selection algorithm to select one best backup route. Only one backup path is selected. In case there are multiple backup path candidates, the selection algorithm selects the best backup path. HFRR provides only two paths, one primary and one backup at any point in time. If the selected backup path itself has two paths in it, then the first path in that backup next hop is used as the backup next hop for the HFRR route.

The primary path is installed with a weight of one. The backup path is installed with a weight of 0x4000. The backup path obviously must be a path through an interface that is not the same as the primary interface.

The backup route is looked up only in the routing table to which interface belongs. For IPv4, the Junos OS uses *routing-instance-name.inet.0*. For IPv6, the Junos OS looks in *routing-instance-name.inet6.0*.

### Characteristics of HFRR Routes

---

The HFRR route is a forwarding-only route and is not used for route resolution. HFRR routes have host addresses, meaning that they have /32 or /128 as the prefix length. In the case of platforms with dual Routing Engines, the backup routing process (rpd) also creates HFRR routes. However, the backup routing process (rpd) does not install HFRR routes to a routing table until the backup becomes the master after a Routing Engine switchover.

Also note that if an HFRR route is present in the routing table, the HFRR route is used for the unicast reverse-path-forwarding (uRPF) computation.

### Removal of HFRR Routes

---

HFRR routes are deleted if the protected interface is deleted or deactivated in the configuration, if HFRR is configured on a routing instance and the routing instance is deactivated or deleted, or when the statement that enables HFRR ([link-protection \(Host Fast Reroute\)](#)) is deleted or deactivated. HFRR routes are deleted and readded when there is a catastrophic operation on routing the instance, such as when the routing process is restarted. HFRR routes are also be removed if all backup routes are deleted. such as when BGP withdraws routes or when BGP is deactivated or deleted.

After a protected interface goes down and if HFRR is deleted or deactivated, a timer starts with a timeout of 20 seconds. The HFRR route deletion occurs after the timer expires. This is to ensure that if the interface is flapping (quickly going up and down), the Junos OS does not unnecessarily perform route deletions and additions that cause traffic loss. This timer is used only when the interface is down or when the HFRR route is deleted or deactivated.

HFRR routes are purged immediately in the following cases:

- A backup route goes down and there are no other potential backup paths.
- An ARP delete message is received.
- The routing process (rpd) terminates.

### Interfaces That Support HFRR

---

HFRR is allowed only on Ethernet interfaces. The commit operation fails if you configure HFRR on point-to-point interfaces.

Only interfaces configured under routing instance of type VPN routing and forwarding (VRF) are accepted. The commit operation fails if you configure HFRR on other types of routing instances.

When the following requirements are not met, the commit operation does not fail. However, the interface is not protected by HFRR, and the interface is marked inactive in the [show hfr profiles](#) command output:

- HFRR is allowed only on numbered interfaces, meaning that an address must be assigned to the interface. You cannot, for example, configure IPv4 on the interface with an address and IPv6 without an address.

- Interfaces that are configured for HFRR protection must be configured at the **[edit interfaces]** hierarchy level and also must be attached to the routing instance.
- The routing instance must have a virtual tunnel (VT) interface or the **vrf-table-label** statement included.

Another reason the interface might be marked inactive in the **show hfr profiles** command output is when the interface is migrating from one instance to another, and the HFRR configuration is in the previous routing instance.

HFRR is not supported on overlapping logical units if they belong to the same routing instance, as shown here:

```
user@host # show interfaces
ge-0/0/2 {
  vlan-tagging;
  unit 0 {
    vlan-id 1;
    family inet {
      address 172.16.0.4/16; # same subnet
    }
  }
  unit 1 {
    vlan-id 2;
    family inet {
      address 172.16.0.5/16; # same subnet
    }
  }
}
```

If you configure overlapping subnets as shown here, and if you enable HFRR on both of the overlapping subnets, the routing protocol process (rpd) generates an RPD\_ASSERT error.

## Example: Configuring Link Protection with Host Fast Reroute

This example shows you how to configure host fast reroute (HFRR). HFRR protects IP endpoints on multipoint interfaces, such as Ethernet.

- [Requirements on page 247](#)
- [Overview on page 248](#)
- [Configuration on page 249](#)
- [Verification on page 255](#)

### Requirements

This example uses the following hardware and software components:

- Two provider edge (PE) devices and four provider (P) devices.
- The example assumes that hosts are present, behind the PE devices.

- The example assumes that at least one Layer 3 switch, such as an EX Series switch, is attached to the hosts.
- Junos OS 11.4R2 or later.

### Overview

---

In this example, traffic flows to server hosts from networks connected to PE devices. This traffic is protected with HFRR. If the link goes down between one PE device and the server farm, traffic is rerouted to the server farm through the other PE device.

You can configure HFRR by adding the **link-protection** statement to the interface configuration in the routing instance.

```
[edit routing-instances cust1 routing-options]  
set interface ge-4/1/0.0 link-protection (Host Fast Reroute)
```

We recommend that you include this statement on all PE devices that are connected to server farms through multipoint interfaces.

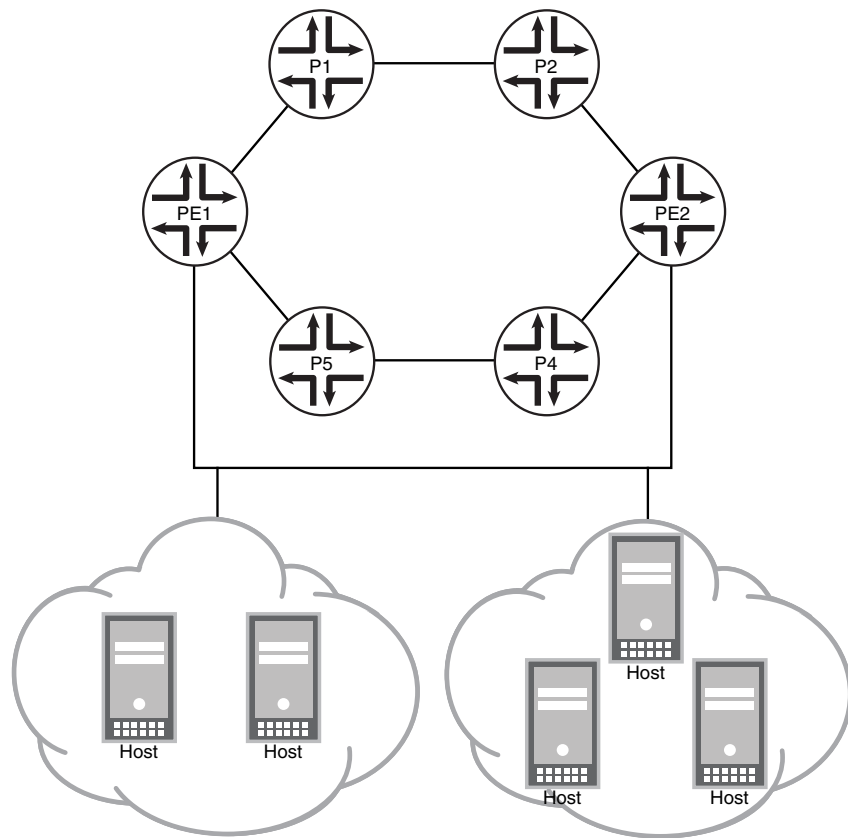
In this example, the PE devices advertise reachability to their server farms through Layer 3 VPN routes and BGP.

As optional settings, the PE devices have the high availability features Nonstop Active Routing and Virtual Router Redundancy Protocol (VRRP) configured. Nonstop active routing (NSR) enables a routing platform with redundant Routing Engines to switch from a primary Routing Engine to a backup Routing Engine without alerting peer nodes that a change has occurred and without losing routing and protocol information. VRRP provides for automatic assignment of available routers to participating hosts, thus increasing the availability and reliability of routing paths.

### Topology Diagram

[Figure 35 on page 249](#) shows the topology used in this example.

Figure 35: Host Fast Reroute Topology



g041180

This example shows the configuration on all of the routing devices and shows the step-by-step procedure on Device PE1.

### Configuration

#### CLI Quick Configuration

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

```
Device PE1
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24
set interfaces ge-4/1/0 unit 0 description toPE2
set interfaces ge-0/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces ge-0/2/0 unit 0 description toP1
set interfaces ge-0/2/0 unit 0 family mpls
set interfaces ge-0/2/4 unit 0 family inet address 10.10.15.2/30
set interfaces ge-0/2/4 unit 0 description toP5
set interfaces ge-0/2/4 unit 0 family mpls
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24 vrrp-group 1 virtual-address 20.20.1.254
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24 vrrp-group 1 priority 240
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24 vrrp-group 1 fast-interval 100
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24 vrrp-group 1 preempt
set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24 vrrp-group 1 accept-data
```

```

set interfaces lo0 unit 0 family inet address 10.255.8.207/32
set protocols mpls interface ge-0/2/0.0
set protocols mpls interface ge-0/2/4.0
set protocols bgp group pe-ce type internal
set protocols bgp group pe-ce local-address 10.255.8.207
set protocols bgp group pe-ce family inet-vpn unicast
set protocols bgp group pe-ce neighbor 10.255.8.86
set protocols bgp group pe-ce export send-routes
set protocols ospf area 0.0.0.0 interface ge-0/2/0.0
set protocols ospf area 0.0.0.0 interface ge-0/2/4.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/2/0.0
set protocols ldp interface ge-0/2/4.0
set policy-options policy-statement send-routes term 1 from protocol direct
set policy-options policy-statement send-routes term 1 from protocol local
set policy-options policy-statement send-routes term 1 then accept
set routing-options nonstop-routing
set routing-options autonomous-system 100
set routing-instances cust1 instance-type vrf
set routing-instances cust1 interface ge-4/1/0.0
set routing-instances cust1 route-distinguisher 100:100
set routing-instances cust1 vrf-target target:100:100
set routing-instances cust1 vrf-table-label
set routing-instances cust1 routing-options interface ge-4/1/0.0 link-protection

```

**Device PE2**

```

set interfaces ge-0/0/2 unit 0 family inet address 10.10.12.2/30
set interfaces ge-0/0/2 unit 0 description toP2
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/1/2 unit 0 family inet address 10.10.13.1/30
set interfaces ge-0/1/2 unit 0 description toP4
set interfaces ge-0/1/2 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 family inet address 20.20.1.2/24
set interfaces ge-2/0/2 unit 0 description toPE1
set interfaces ge-2/0/2 unit 0 family inet address 20.20.1.2/24 vrrp-group 1 virtual-address
20.20.1.254
set interfaces ge-2/0/2 unit 0 family inet address 20.20.1.2/24 vrrp-group 1 fast-interval
100
set interfaces ge-2/0/2 unit 0 family inet address 20.20.1.2/24 vrrp-group 1 preempt
set interfaces ge-2/0/2 unit 0 family inet address 20.20.1.2/24 vrrp-group 1 accept-data
set interfaces lo0 unit 0 family inet address 10.255.8.86/32
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/1/2.0
set protocols bgp group pe-ce type internal
set protocols bgp group pe-ce export send-routes
set protocols bgp group pe-ce local-address 10.255.8.86
set protocols bgp group pe-ce family inet-vpn unicast
set protocols bgp group pe-ce neighbor 10.255.8.207
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface ge-0/1/2.0
set policy-options policy-statement send-routes term 1 from protocol direct
set policy-options policy-statement send-routes term 1 from protocol local
set policy-options policy-statement send-routes term 1 then accept
set routing-options nonstop-routing

```

```

set routing-options autonomous-system 100
set routing-instances cust1 instance-type vrf
set routing-instances cust1 interface ge-2/0/2.0
set routing-instances cust1 route-distinguisher 100:100
set routing-instances cust1 vrf-target target:100:100
set routing-instances cust1 vrf-table-label
set routing-instances cust1 routing-options interface ge-2/0/2.0 link-protection

```

Device P1

```

set interfaces ge-0/0/3 unit 0 family inet address 10.10.11.1/30
set interfaces ge-0/0/3 unit 0 description toP2
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 unit 0 family inet address 10.10.10.2/30
set interfaces ge-0/0/4 unit 0 description toPE1
set interfaces ge-0/0/4 unit 0 family mpls
set protocols mpls interface ge-0/0/4.0
set protocols mpls interface ge-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/0/3.0
set protocols ldp interface ge-0/0/4.0
set routing-options autonomous-system 100

```

Device P2

```

set interfaces ge-0/2/1 unit 0 family inet address 10.10.12.1/30
set interfaces ge-0/2/1 unit 0 description toPE2
set interfaces ge-0/2/1 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.10.11.2/30
set interfaces ge-1/2/1 unit 0 description toP1
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.246/32
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 100

```

Device P4

```

set interfaces ge-0/2/3 unit 0 family inet address 10.10.13.2/30
set interfaces ge-0/2/3 unit 0 description toPE2
set interfaces ge-0/2/3 unit 0 family mpls
set interfaces ge-1/3/3 unit 0 family inet address 10.10.14.1/30
set interfaces ge-1/3/3 unit 0 description toP5
set interfaces ge-1/3/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.4/32
set protocols mpls interface ge-0/2/3.0
set protocols mpls interface ge-1/3/3.0
set protocols ospf area 0.0.0.0 interface ge-0/2/3.0
set protocols ospf area 0.0.0.0 interface ge-1/3/3.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/2/3.0
set protocols ldp interface ge-1/3/3.0
set routing-options autonomous-system 100

```

Device P5

```

set interfaces ge-0/1/2 unit 0 family inet address 10.10.15.1/30

```

```

set interfaces ge-0/1/2 unit 0 description toPE1
set interfaces ge-0/1/2 unit 0 family mpls
set interfaces ge-0/1/5 unit 0 family inet address 10.10.14.2/30
set interfaces ge-0/1/5 unit 0 description toP4
set interfaces ge-0/1/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.5/32
set protocols mpls interface ge-0/1/5.0
set protocols mpls interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface ge-0/1/5.0
set protocols ospf area 0.0.0.0 interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/1/2.0
set protocols ldp interface ge-0/1/5.0
set routing-options autonomous-system 100

```

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

To configure HFRR:

1. Configure the interfaces.

```

[edit interfaces]
user@PE1# set interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24
user@PE1# set interfaces ge-4/1/0 unit 0 description toPE2
user@PE1# set interfaces ge-0/2/0 unit 0 family inet address 10.10.10.1/30
user@PE1# set interfaces ge-0/2/0 unit 0 description toP1
user@PE1# set interfaces ge-0/2/0 unit 0 family mpls
user@PE1# set interfaces ge-0/2/4 unit 0 family inet address 10.10.15.2/30
user@PE1# set interfaces ge-0/2/4 unit 0 description toP5
user@PE1# set interfaces ge-0/2/4 unit 0 family mpls
user@PE1# set interfaces lo0 unit 0 family inet address 10.255.8.207/32

```

2. (Optional) Configure VRRP on the interface to Device PE2.

```

[edit interfaces ge-4/1/0 unit 0 family inet address 20.20.1.1/24]
set vrrp-group 1 virtual-address 20.20.1.254
set vrrp-group 1 priority 240
set vrrp-group 1 fast-interval 100
set vrrp-group 1 preempt
set vrrp-group 1 accept-data

```

3. Configure MPLS on the interfaces.

```

[edit protocols mpls]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0

```

4. Configure BGP.

```

[edit protocols bgp group pe-ce]
user@PE1# set type internal
user@PE1# set local-address 10.255.8.207
user@PE1# set family inet-vpn unicast
user@PE1# set neighbor 10.255.8.86
user@PE1# set export send-routes

```

5. Configure a policy that advertises direct and local interface routes.
 

```
[edit policy-options policy-statement send-routes term 1]
user@PE1# set from protocol direct
user@PE1# set from protocol local
user@PE1# set then accept
```
6. Configure an interior gateway protocol, such as IS-IS or OSPF.
 

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0
user@PE1# set interface lo0.0 passive
```
7. Configure a signaling protocol, such as RSVP or LDP.
 

```
[edit protocols ldp]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0
```
8. (Optional) Configure nonstop active routing.
 

```
[edit routing-options]
user@PE1# set nonstop-routing
```
9. Configure the autonomous system (AS).
 

```
[edit routing-options]
user@PE1# set routing-options autonomous-system 100
```
10. Configure the Layer 3 VPN routing instance.
 

```
[edit routing-instances cust1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-4/1/0.0
user@PE1# set route-distinguisher 100:100
user@PE1# set vrf-target target:100:100
user@PE1# set vrf-table-label
```
11. Configure HFRR link protection.
 

```
[edit routing-instances cust1 routing-options]
user@PE1# set interface ge-4/1/0.0 link-protection (Host Fast Reroute)
```
12. If you are done configuring the device, commit the configuration.
 

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

### Results

Confirm your configuration by issuing the **show interfaces**, **show protocols**, **show policy-options**, **show routing-options**, and **show routing-instances** commands.

```
user@PE1# show interfaces
ge-4/1/0 {
  unit 0 {
    description toPE2;
    family inet {
      address 20.20.1.1/24 {
        vrrp-group 1 {
```

```
        virtual-address 20.20.1.254;
        priority 240;
        fast-interval 100;
        preempt;
        accept-data;
    }
}
}
}
}
ge-0/2/0 {
    unit 0 {
        description toP1;
        family inet {
            address 10.10.10.1/30;
        }
        family mpls;
    }
}
ge-0/2/4 {
    unit 0 {
        description toP5;
        family inet {
            address 10.10.15.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.8.207/32;
        }
    }
}

user@PE1# show protocols
mpls {
    interface ge-0/2/0.0;
    interface ge-0/2/4.0;
}
bgp {
    group pe-ce {
        export-send-routes;
        type internal;
        local-address 10.255.8.207;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.8.86;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/2/0.0;
        interface ge-0/2/4.0;
```

```

        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-0/2/0.0;
    interface ge-0/2/4.0;
}

user@PE1# show policy-options
policy-statement send-routes {
    term 1 {
        from protocol [ direct local ];
        then accept;
    }
}

user@PE1# show routing-options
nonstop-routing;
autonomous-system 100;

user@PE1# show routing-instances
cust1 {
    instance-type vrf;
    interface ge-4/1/0.0;
    route-distinguisher 100:100;
    vrf-target target:100:100;
    vrf-table-label;
    routing-options {
        interface {
            ge-4/1/0.0 {
                link-protection;
            }
        }
    }
}

```

## Verification

Confirm that the configuration is working properly.

- [Verifying HFRR on page 255](#)
- [Verifying ARP Routes on page 256](#)
- [Verifying Fast Reroute Routes on page 256](#)
- [Verifying Forwarding on page 257](#)

### Verifying HFRR

**Purpose** Make sure that HFRR is enabled.

**Action** user@PE1> `show hfr profiles`  
 HFRR pointer: 0x9250000  
 HFRR Current State: HFRR\_ACTIVE  
 HFRR Protected IFL Name: ge-4/1/0.0  
 HFRR Protected IFL Handle: 0x921086c  
 HFRR Routing Instance Name: cust1  
 HFRR Routing Instance Handle: 0x9129740  
 HFRR Sync BG Scheduled: NO  
 HFRR RTS Filter On: YES  
 HFRR Delete BG Scheduled: NO  
 HFRR Num ARP Routes learnt: 100  
 HFRR Num FRR Routes Created: 100

**Meaning** The output shows that the HFRR is enabled on interface ge-4/1/0.0.

### *Verifying ARP Routes*

**Purpose** Make sure that the expected ARP routes are learned.

**Action** user@PE1> `show route protocol arp`  
 inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)  
  
 inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)  
  
 cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)  
 + = Active Route, - = Last Active, \* = Both

20.20.1.3/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.4/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.5/32	@[ARP/4294967293] 00:04:32, from 20.20.1.1 Unusable
20.20.1.6/32	@[ARP/4294967293] 00:04:34, from 20.20.1.1 Unusable
20.20.1.7/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.8/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.9/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.10/32	@[ARP/4294967293] 00:04:35, from 20.20.1.1 Unusable
20.20.1.11/32	@[ARP/4294967293] 00:04:33, from 20.20.1.1 Unusable
20.20.1.12/32	@[ARP/4294967293] 00:04:33, from 20.20.1.1 Unusable
20.20.1.13/32	@[ARP/4294967293] 00:04:33, from 20.20.1.1 Unusable
...	

### *Verifying Fast Reroute Routes*

**Purpose** Make sure that the expected fast reroute (FRR) routes are learned.

**Action** user@PE1> show route protocol frr

```
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.20.1.3/32      #[FRR/200] 00:05:38, from 20.20.1.1
                  > to 20.20.1.3 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.4/32      #[FRR/200] 00:05:38, from 20.20.1.1
                  > to 20.20.1.4 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.5/32      #[FRR/200] 00:05:35, from 20.20.1.1
                  > to 20.20.1.5 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.6/32      #[FRR/200] 00:05:37, from 20.20.1.1
                  > to 20.20.1.6 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.7/32      #[FRR/200] 00:05:38, from 20.20.1.1
                  > to 20.20.1.7 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.8/32      #[FRR/200] 00:05:38, from 20.20.1.1
                  > to 20.20.1.8 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.9/32      #[FRR/200] 00:05:38, from 20.20.1.1
                  > to 20.20.1.9 via ge-4/1/0.0
                  to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.10/32     #[FRR/200] 00:05:38, from 20.20.1.1
...

```

#### *Verifying Forwarding*

**Purpose** Make sure that the expected routes appear in the forwarding table.

```

Action user@PE1> show route forwarding-table destination 20.20.1.3
Routing table: default.inet
Internet:
Destination      Type RtRef Next hop      Type Index NhRef Netif
default          perm  0                               rjct  36   1

Routing table: default-switch.inet
Internet:
Destination      Type RtRef Next hop      Type Index NhRef Netif
default          perm  0                               rjct  554  1

Routing table: __master.anon__.inet
Internet:
Destination      Type RtRef Next hop      Type Index NhRef Netif
default          perm  0                               rjct  532  1

Routing table: cust1.inet
Internet:
Destination      Type RtRef Next hop      Type Index NhRef Netif
20.20.1.3/32      user  0                               ulst 1048575 2
                                     0:0:14:14:1:3 ucst  767   3 ge-4/1/0.0
                                     10.10.15.1  indr 1048574 1001
                                     Push 16, Push 299792(top) 1262
      2 ge-0/2/4.0
20.20.1.3/32      dest  0 0:0:14:14:1:3 ucst  767   3 ge-4/1/0.0

...

```

**Related Documentation**

- [Fast Reroute Overview](#)

## Example: Configuring a Layer 3 VPN with Route Reflection and AS Override

Suppose that you are a service provider providing a managed MPLS-based Layer 3 VPN service. Your customer has several sites and requires BGP routing to customer edge (CE) devices at each site.

- [Requirements on page 258](#)
- [Overview on page 258](#)
- [Configuration on page 259](#)
- [Verification on page 266](#)

### Requirements

No special configuration beyond device initialization is required before configuring this example.

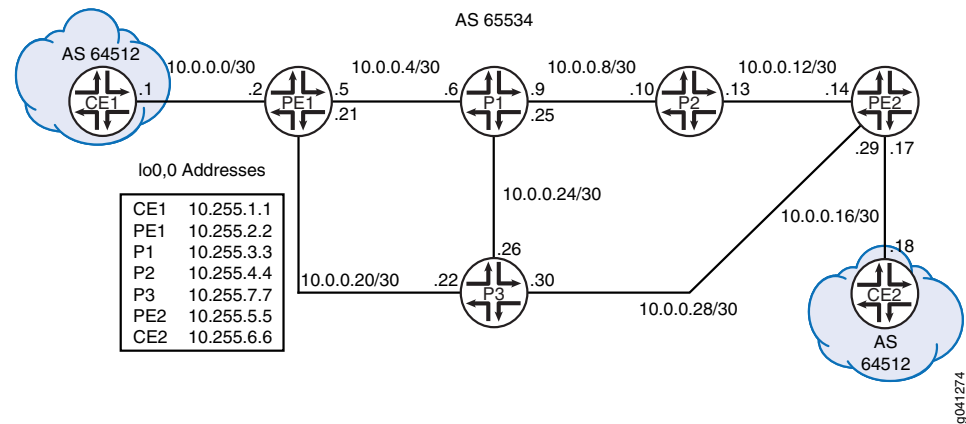
### Overview

This example has two CE devices, two provider edge (PE) devices, and several provider core devices. The provider network is also using IS-IS to support LDP and BGP loopback reachability. Device P2 is acting as a route reflector (RR). Both CE devices are in autonomous system (AS) 64512. The provider network is in AS 65534.

The **as-override** statement is applied to the PE devices, thus replacing the CE device's AS number with that of the PE device. This prevents the customer AS number from appearing more than once in the AS path attribute.

Figure 36 on page 259 shows the topology used in this example.

Figure 36: AS Override Topology



"CLI Quick Configuration" on page 259 shows the configuration for all of the devices in Figure 36 on page 259. The section "Step-by-Step Procedure" on page 263 describes the steps on Device PE1.

## Configuration

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

**Device CE1**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.1.1/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0101.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.2
set policy-options policy-statement ToBGP term Direct from protocol direct
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.1.1
set routing-options autonomous-system 64512

```

**Device P1**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.6/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.25/30

```

```

set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.3.3/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0303.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.3.3
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.3.3

```

**Device P2**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.10/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.13/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.4.4/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0404.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group Core-RRClients type internal
set protocols bgp group Core-RRClients local-address 10.255.4.4
set protocols bgp group Core-RRClients family inet-vpn unicast
set protocols bgp group Core-RRClients cluster 10.255.4.4
set protocols bgp group Core-RRClients peer-as 65534
set protocols bgp group Core-RRClients neighbor 10.255.3.3
set protocols bgp group Core-RRClients neighbor 10.255.7.7
set protocols bgp group Core-RRClients neighbor 10.255.2.2
set protocols bgp group Core-RRClients neighbor 10.255.5.5
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.4.4
set routing-options autonomous-system 65534

```

**Device P3**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.22/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.26/30
set interfaces ge-1/2/1 unit 0 family iso

```

```

set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.30/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.7.7/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0707.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.7.7
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.7.7

```

#### Device PE1

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.21/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.2.2/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0202.00
set protocols mpls interface ge-1/2/2.0
set protocols mpls interface ge-1/2/1.0
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.2.2
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/1.0 level 2 metric 10
set protocols isis interface ge-1/2/1.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf

```

```

set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 as-override
set routing-options router-id 10.255.2.2
set routing-options autonomous-system 65534

```

**Device PE2**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.14/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.17/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.29/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.5.5/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0505.00
set protocols mpls interface ge-1/2/0.0
set protocols mpls interface ge-1/2/2.0
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.5.5
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/0.0 level 2 metric 10
set protocols isis interface ge-1/2/0.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/0.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/1.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 as-override
set routing-options router-id 10.255.5.5
set routing-options autonomous-system 65534

```

**Device CE2**

```

set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.18/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.6.6/32

```

```

set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0606.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.17
set policy-options policy-statement ToBGP term Direct from protocol direct
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.6.6
set routing-options autonomous-system 64512

```

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

To configure AS override:

1. Configure the interfaces.

To enable MPLS, include the protocol family on the interface so that the interface does not discard incoming MPLS traffic.

```

[edit interfaces]
user@PE1# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
user@PE1# set ge-1/2/0 unit 0 family iso
user@PE1# set ge-1/2/0 unit 0 family mpls
user@PE1# set ge-1/2/1 unit 0 family inet address 10.0.0.5/30
user@PE1# set ge-1/2/1 unit 0 family iso
user@PE1# set ge-1/2/1 unit 0 family mpls
user@PE1# set ge-1/2/2 unit 0 family inet address 10.0.0.21/30
user@PE1# set ge-1/2/2 unit 0 family iso
user@PE1# set ge-1/2/2 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.255.2.2/32
user@PE1# set lo0 unit 0 family iso address 49.0001.0010.0000.0202.00

```

2. Add the interface to the MPLS protocol to establish the control plane level connectivity.

Set up the IGP so that the provider devices can communicate with each other.

To establish a mechanism to distribute MPLS labels, enable LDP. Optionally, for LDP, enable forwarding equivalence class (FEC) deaggregation, which results in faster global convergence.

```

[edit protocols]
user@PE1# set mpls interface ge-1/2/2.0
user@PE1# set mpls interface ge-1/2/1.0
user@PE1# set mpls interface lo0.0
user@PE1# set mpls interface fxp0.0 disable
user@PE1# set isis interface ge-1/2/1.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/1.0 level 1 disable
user@PE1# set isis interface ge-1/2/2.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/2.0 level 1 disable
user@PE1# set isis interface fxp0.0 disable
user@PE1# set isis interface lo0.0 level 2 metric 0
user@PE1# set ldp deaggregate
user@PE1# set ldp interface ge-1/2/1.0

```

```

user@PE1# set ldp interface ge-1/2/2.0
user@PE1# set ldp interface fxp0.0 disable
user@PE1# set ldp interface lo0.0

```

3. Enable the internal BGP (IBGP) connection to peer with the RR only, using the IPv4 VPN unicast address family.

```

[edit protocols bgp group l3vpn]
user@PE1# set type internal
user@PE1# set local-address 10.255.2.2
user@PE1# set family inet-vpn unicast
user@PE1# set peer-as 65534
user@PE1# set local-as 65534
user@PE1# set neighbor 10.255.4.4

```

4. Configure the routing instance, including the **as-override** statement.

Create the routing-instance (VRF) on the PE device, setting up the BGP configuration to peer with Device CE1.

```

[edit routing-instances VPN-A]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 65534:1234
user@PE1# set vrf-target target:65534:1234
user@PE1# set protocols bgp group CE type external
user@PE1# set protocols bgp group CE family inet unicast
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 as-override

```

5. Configure the router ID and the AS number.

```

[edit routing-options]
user@PE1# set router-id 10.255.2.2
user@PE1# set autonomous-system 65534

```

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

```

user@PE1# show interfaces
ge-1/2/0 {
  unit 2 {
    family inet {
      address 10.0.0.2/30;
    }
    family iso;
    family mpls;
  }
}
ge-1/2/1 {
  unit 5 {
    family inet {
      address 10.0.0.5/30;
    }
    family iso;
  }
}

```

```

        family mpls;
    }
}
ge-1/2/2 {
    unit 21 {
        family inet {
            address 10.0.0.21/30;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.2.2/32;
        }
        family iso {
            address 49.0001.0010.0000.0202.00;
        }
    }
}
}

user@PE1# show protocols
mpls {
    interface ge-1/2/2.0;
    interface ge-1/2/1.0;
    interface lo0.0;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group l3vpn {
        type internal;
        local-address 10.255.2.2;
        family inet-vpn {
            unicast;
        }
        peer-as 65534;
        local-as 65534;
        neighbor 10.255.4.4;
    }
}
isis {
    interface ge-1/2/1.0 {
        level 2 metric 10;
        level 1 disable;
    }
    interface ge-1/2/2.0 {
        level 2 metric 10;
        level 1 disable;
    }
    interface fxp0.0 {
        disable;
    }
}

```

```

interface lo0.0 {
    level 2 metric 0;
}
}
ldp {
    deaggregate;
    interface ge-1/2/1.0;
    interface ge-1/2/2.0;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}

user@PE1# show routing-instances
VPN-A {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 65534:1234;
    vrf-target target:65534:1234;
    protocols {
        bgp {
            group CE {
                type external;
                family inet {
                    unicast;
                }
                neighbor 10.0.0.1 {
                    peer-as 64512;
                    as-override;
                }
            }
        }
    }
}

user@PE1# show routing-options
router-id 10.255.2.2;
autonomous-system 65534;

```

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

## Verification

Confirm that the configuration is working properly.

- [Checking AS Path to the CE Devices on page 266](#)
- [Checking How the Route to Device CE2 Is Advertised on page 267](#)
- [Checking the Route on Device CE1 on page 267](#)

### Checking AS Path to the CE Devices

**Purpose** Display information on Device PE1 about the AS path attribute for the route to Device CE2's loopback interface.

**Action** On Device PE1, from operational mode, enter the **show route table VPN-A.inet.0 10.255.6.6** command.

```
user@PE1> show route table VPN-A.inet.0 10.255.6.6
```

```
VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
10.255.6.6/32      *[BGP/170] 02:19:35, localpref 100, from 10.255.4.4
                   AS path: 64512 I, validation-state: unverified
                   > to 10.0.0.22 via ge-1/2/2.0, Push 300032, Push 299776(top)
```

**Meaning** The output shows that Device PE1 has an AS path for 10.255.6.6/32 as coming from AS 64512.

### Checking How the Route to Device CE2 Is Advertised

**Purpose** Make sure the route to Device CE2 is advertised to Device CE1 as if it is coming from the MPLS core.

**Action** On Device PE1, from operational mode, enter the **show route advertising-protocol bgp 10.0.0.1** command.

```
user@PE1> show route advertising-protocol bgp 10.0.0.1
```

```
VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED      Lclpref    AS path
* 10.0.0.16/30          Self                    MED         I
* 10.255.1.1/32         10.0.0.1              65534 I
* 10.255.6.6/32         Self                    65534 I
```

**Meaning** The output indicates that Device PE1 is advertising only its own AS number in the AS path.

### Checking the Route on Device CE1

**Purpose** Make sure that Device CE1 contains only the provider AS number in the AS path for the route to Device CE2.

**Action** From operational mode, enter the **show route table inet.0 terse 10.255.6.6** command.

```
user@CE1> show route table inet.0 terse 10.255.6.6
```

```
inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
```

A	V	Destination	P	Prf	Metric 1	Metric 2	Next hop	AS path
*	?	10.255.6.6/32	B	170	100			65534 65534
		I						
		unverified					>10.0.0.2	

**Meaning** The output indicates that Device CE1 has a route to Device CE2. The loop issue is resolved with the use of the **as-override** statement.

One route is hidden on the CE device. This is because Junos OS does not perform a BGP split horizon. Generally, split horizon in BGP is unnecessary, because any routes that might be received back by the originator are less preferred due to AS path length (for EBGP), AS path loop detection (IBGP), or other BGP metrics. Advertising routes back to the neighbor from which they were learned has a negligible effect on the router's performance, and is the correct thing to do.

- Related Documentation**
- *Understanding AS Override*
  - *Example: Disabling Suppression of Route Advertisements*

---

## Example: Configuring MPLS Egress Protection for Layer 3 VPN Services

---

This example describes a local repair mechanism for protecting Layer 3 VPN services against egress provider edge (PE) router failure in a scenario where the customer edge (CE) routers are multihomed with more than one PE router.

The following terminology is used in this example:

- **Originator PE router**—A PE router with protected routing instances or subnets that distributes the primary Layer 3 VPN route.
- **Backup PE router**—A PE router that announces a backup Layer 3 VPN route.
- **Protector PE router**—A router that cross-connects VPN labels distributed by the originator PE router to the labels originated by the backup PE router. The protector PE router can also be a backup PE router.
- **Transport LSP**—An LDP-signaled label-switched path (LSP) for BGP next hops.
- **PLR**—A router acting as the point of local repair (PLR) that can redirect Layer 3 VPN traffic to a protector PE router to enable fast restoration and reroute.
- **Loop-free alternate routes**—A technology that essentially adds IP fast-reroute capability for the interior gateway protocol (IGP) by precomputing backup routes for all the primary routes of the IGP. In the context of this document, the IGP is IS-IS.
- **Multihoming**—A technology that enables you to connect a CE device to multiple PE routers. In the event that a connection to the primary PE router fails, traffic can be automatically switched to the backup PE router.
- **Context identifier**—An IPv4 address used to identify the VPN prefix that requires protection. The identifier is propagated to the PE and PLR core routers, making it possible for the protected egress PE router to signal the egress protection to the protector PE router.
- **Dual protection**—A protection mechanism where two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops. For example, between the two PE routers PE1 and PE2, PE1 could be a primary PE router for context identifier 3.1.0.0 and protector for context identifier 4.1.0.0. Likewise, the PE2 router could be a protector for context identifier 3.1.0.0 and a primary PE router for context identifier 4.1.0.0.

This example contains the following topics:

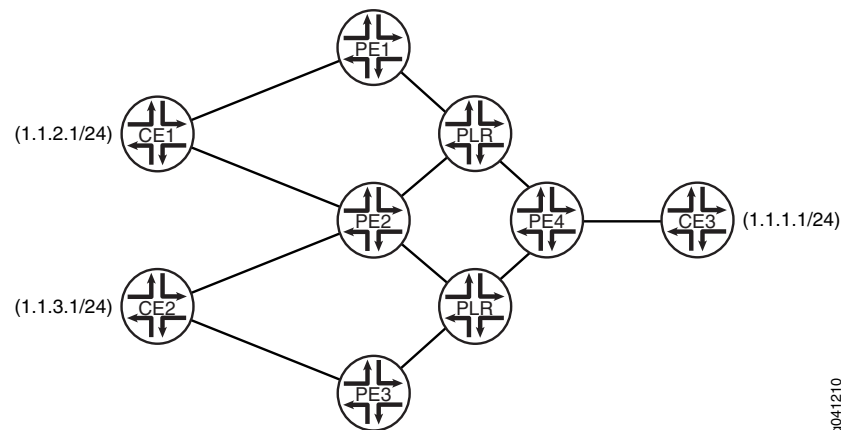
- [Egress Protection for Layer 3 VPN Edge Protection Overview on page 269](#)
- [Example: Configuring Egress Protection for Layer 3 VPN Services on page 275](#)

## Egress Protection for Layer 3 VPN Edge Protection Overview

Typically, Layer 3 VPN service restoration for multihomed customer edge (CE) routers depends on the ingress provider edge (PE) router to detect the egress PE link or node failure and switch traffic to the backup PE router. To achieve faster restoration, a protector mechanism for the PE router can be used to perform local restoration of the service immediately in case of an egress PE node failure. This mechanism requires the router at the point of local repair (PLR) to redirect VPN traffic to a protector PE router for fast reroute of traffic.

The following topology describes the concept of egress protection.

**Figure 37: Sample Topology for Egress Protection**



In this topology:

Router PE3 acts as the protector for the PE2 Layer 3 VPN routing instances or subnets.

The CE routers are part of a VPN where Router CE1 is multihomed with Router PE1 and Router PE2. Likewise, Router CE2 is multihomed with Routers PE2 and PE3.

Router PE1 can be the originator for the context identifier for Router CE1, while Router PE2 is the protector for that context identifier. Likewise, PE2 can be the originator for the context identifier for Router CE2, while Router PE3 is the protector for that context identifier.

The working path taken by Router PE4 might be through PLR>PE2 for both Router CE1 and Router CE2. The backup path for Router CE1 is through PLR>PE1. The backup path for Router CE2 is through PLR>PE3. Traffic flows through the working path under normal circumstances.

When Router PE4 detects a PE2 node or link failure, traffic is rerouted from the working path to the protected path. In the normal failover process, the detection of failure and the recovery rely on the control plane and is therefore relatively slow.

Typically, if there is a link or node failure in the core network, the egress PE router would have to rely on the ingress PE router to detect the failure and switch over to the backup path, because a local repair option for egress failure is not available.

To provide a local repair solution for the egress PE link or node failure, a mechanism known as egress protection can be used to repair and restore the connection quickly. If egress protection is configured, the PLR router detects the PE2 link or node failure and reroutes traffic through the protector Router PE3 using the backup LDP-signaled label-switched path (LSP). The PLR router uses per-prefix loop-free alternate routes to program the backup next hop through Router PE3, and traffic is forwarded to Routers CE1 and CE2 using the alternate paths. This restoration is done quickly after the PLR router detects the Router PE2 egress node or link failure.

The dual protection mechanism can also be used for egress protection where the two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops.

---

## Router Functions

In [Figure 37 on page 269](#), the following routers perform the following functions:

### ***Protected PE Router***

The protected PE, PE2, performs the following functions:

- Updates a context identifier for the BGP next hop for the Layer 3 VPN prefix.
- Advertises the context identifier to the IS-IS domain.

### ***Protector PE Router***

The protector PE router, PE3, performs the following functions:

- Advertises the context identifier to the IS-IS domain with a high metric. The high IGP metric (configurable) along with the LDP label ensures that the PLR router uses the LDP-signaled backup LSP in the event of an egress PE router failure.
- Builds a context-label table for route lookup and a backup forwarding table for the protected PE router (PE2).



**NOTE:** The protector PE router should not be in the forwarding path to the primary PE router.

---

### PLR Router

The router acting as the point of local repair (PLR) performs the following functions:

- Computes per-prefix loop-free alternate routes. For this computation to work, the configuration of the **node-link-protection** statement and the **backup-spf-options per-prefix-calculation** statement is necessary at the **[edit protocols isis]** hierarchy level.
- Installs backup next hops for the context identifier through the PE3 router (protector PE).
- Detects PE router failure and redirects the transport LSP traffic to the protector.



**NOTE:** The PLR router must be directly connected to the protector router (in this case, PE3). If not, the loop-free alternate route cannot find the backup path to the protector. This limitation is removed in Junos OS Release 13.3 and later.

### Protector and Protection Models

Protector is a new role or function for the restoration of egress PE node failure. This role could be played by a backup egress PE router or any other node that participates in the VPN control plane for VPN prefixes that require egress node protection. There are two protection models based on the location and role of a protector:

- **Co-located protector**—In this model, the protector PE router and the backup PE router configurations are done on the same router. The protector is co-located with the backup PE router for the protected prefix, and it has a direct connection to the multihomed site that originates the protected prefix. In the event of an egress PE failure, the protector receives traffic from the PLR router and routes the traffic to the multihomed site.
- **Centralized protector**—In this model, the protector PE router and the backup PE router are different. The centralized protector might not have a direct connection to the multihomed site. In the event of an egress PE link or node failure, the centralized protector reroutes the traffic to the backup egress PE router with the VPN label advertised for the backup egress PE router that takes over the role of sending traffic to the multihomed site.

A network can use either of the protection models or a combination of both, depending on the requirement.

For more information about egress PE failure protection, see Internet draft [draft-minto-2547-egress-node-fast-protection-00](#), *2547 egress PE Fast Failure Protection*.

### IGP Advertisement Model

Egress protection availability is advertised in the interior gateway protocol (IGP). Label protocols along with Constrained Shortest Path First (CSPF) use this information to do egress protection.

For Layer 3 VPNs, the IGP advertisements can be of the following types:

- Context identifier as a stub link (supported in Junos OS 11.4R3 and later). A link connecting a stub node to a transit node is a stub link.
- Context identifier as a stub alias node (supported in Junos OS 13.3 and later).
- Context identifier as a stub proxy node (supported in Junos OS 13.3 and later).

By default, the stub link is used. To enable enhanced point-of-local-repair (PLR) functionality, in which the PLR reroutes service traffic during an egress failure, configure a stub alias node or a stub proxy node as follows:

```
[edit protocols mpls egress-protection context-identifier 6.6.6.6]
user@host# set advertise-mode ?
Possible completions:
  stub-alias      Alias
  stub-proxy      Proxy
```

The two methods offer different advantages, depending on the needs of your network deployment.

### ***Context Identifier as a Stub Alias Node***

In the stub alias method, the LSP end-point address has an explicit backup egress node where the backup can be learned or configured on the penultimate hop node of a protected LSP. With this model, the penultimate hop node of a protected LSP sets up the bypass LSP tunnel to back up the egress node by avoiding the primary egress node. This model requires a Junos OS upgrade in core nodes, but is flexible enough to support all traffic engineering constraints.

The PLR learns that the context ID has a protector. When the primary context ID goes down, packets are rerouted to the protector by way of a pre-programmed backup path. The context ID and protector mapping are configured or learned on the PLR and signaled in the IGP from the protector. A routing table called inet.5 on the PLR provides the configured or IGP-learned details.

IS-IS advertises context IDs into the TED through an IP address TLV. IS-IS imports this TLV into the TED as extended information. IS-IS advertises the protector TLV routes in the inet.5 route for the context ID with protocol next hop being the protector's router ID. If the protector TLV has a label, the label is added to the route in the inet.5 routing table for LDP to use.

CSPF considers the IP address TLV for tunnel endpoint computation.

With the stub alias model, the protector LSP setup does not require any changes in any nodes. But bypass LSP setup for node protection requires changes in the PHN and the protector router.

When RSVP sets up bypass for node protection LSP, RSVP also performs a lookup for the protector if the PLR is the penultimate hop of the LSP. If the protector is available for the LSP destination, it uses CSPF to compute a path with a constraint that excludes the egress PE and sets up a bypass LSP destination to the context ID if one is not already set up. When setting up a bypass LSP to the context ID, the PLR unsets all protection options.

LDP is useful in the case when the network supports 100 percent LFA coverage but does not support 100 percent per-prefix LFA coverage. LDP sets up a backup path with the protector with the context label advertised by the protector to the service point.

In networks in which 100 percent LFA coverage is not available, it is useful to have backup LSP LFAs with RSVP-based tunnels.

In a steady state, the forwarding is the same as on any other protected LSP in the PLR. In the protector, the non-null label that is advertised and signaled for the context ID has the table next hop point to the MPLS context table, where the peers' labels are programmed.

During a failure, the PLR swaps the transport label with the bypass LSP for the context ID or swaps the label context-label (the protector-advertised label for the context ID) and pushes the transport label to the protector lo0 interface address.

### ***Context Identifier as a Stub Proxy Node***

Context identifier as a stub proxy node (supported in Junos OS 13.3 and later). A stub node is one that only appears at the end of an AS path, which means it does not provide transit service. In this mode, known as the virtual or proxy mode, the LSP end-point address is represented as a node with bidirectional links, with the LSP's primary egress node and backup egress node. With this representation, the penultimate hop of the LSP primary egress point can behave like a PLR in setting up a bypass tunnel to back up the egress by avoiding the primary egress node. This model has the advantage that you do not need to upgrade Junos OS on core nodes and will thereby help operators to deploy this technology.

The context ID is represented as a node in the traffic engineering (TE) and IGP databases. The primary PE device advertises the context node into the IGP and TE databases. The primary PE device and the protected PE device support one link to the context node with a bandwidth and a TE metric. Other TE characteristics of TE links are not advertised by Junos OS.

In IS-IS, the primary PE router advertises the proxy node along with links to the primary router and the protector router. The primary and the protector routers advertise links to the proxy node. The proxy node builds the following information.

- System ID—Binary-coded decimal based on the context ID.
- Host name—Protector-name:context ID
- LSP-ID—<System-ID>.00
- PDU type—Level 2 and Level 1, based on the configuration
- LSP attributes:
  - Overload—1
  - IS\_TYPE\_L1(0x01) | IS\_TYPE\_L2(0x02) for the level 2 PDU
  - IS\_TYPE\_L1 for level 1

- Multiarea—No
- All other attributes—0

The proxy node only contains area, MT, host name, router ID, protocols and IS reachability TLVs. The area, MT, authentication, and protocols TLV are the same as on the primary. The IS reachability TLVs contains two links called Cnode-primary-link and Cnode-protector-link. Both links include TE TLVs. The following TE-link-TLVs are advertised in context links:

- IPv4 interface or neighbor address
- Maximum bandwidth
- TE default metric
- Link (local or remote) Identifiers

Sub TLV values:

- Bandwidth—zero
- TE metric—Maximum TE metric
- Interface address—context ID
- Protector neighbor address—protector router ID
- Primary neighbor address—protected router ID
- Link local-ID protector—0x80ffff1
- Link local-ID primary—0x80ffff2
- Link remote-ID protector—Learned from protector
- Link remote-ID primary—Learned from primary

Protected PE links to context node (primary advertises the link with the following details):

- Bandwidth—Maximum
- TE metric—1
- Interface address—Router ID
- Context neighbor address—Context ID
- Link local-ID to context node—Automatically generated (similar to a sham link)
- Link remote-ID to context node—0x80ffff2

Protector PE links to context node:

- The protector advertises unnumbered transit links with the maximum routable link metric and the maximum TE metric and zero bandwidth to the context node. Other TE characteristics are not advertised.

Unnumbered links are advertised with the following attributes:

- bandwidth—0
- TE metric—MAX TE metric
- Interface address—Router ID
- Context neighbor address—Context ID
- Link local ID to context node—Autogenerated (similar to a sham link)
- Link remote ID to context node—0x80fffff1

In RSVP, the behavior changes are only in the protector and primary routers. RSVP terminates the LSP and the bypass LSP to the context ID. If the context ID is the protector, a non-null label is signaled. Otherwise, it will be based on the configuration or the requested label type. RSVP verifies the Explicit Route Object (ERO) from the path for itself and the context ID. RSVP sends the Resv message with two Record Route Object (RRO) objects—one for the context ID and one for itself. This simulates the penultimate-hop node (PHN) to do node protection with the protector for the primary for context ID LSP. As the fast reroute (FRR)-required bypass, the LSP has to merge back to the protector LSP PHN setup bypass to context ID through the protector by avoiding the primary.

The protector also terminates the backup LSP for the context ID to keep the protected LSP alive during a failure until the ingress node resigns the LSP. The new LSP is reestablished through the protector, but this LSP is not used for service traffic as service protocol does not use the context ID. The LSP traverses through the protector even if the primary comes up. Only reoptimization resigns the LSP through the primary. In stub proxy mode, the bypass LSP with constraints is not supported.

LDP cannot use the stub proxy method due to the inflated metric advertised in the IGP.

With regard the forwarding state, a PE router that protects one or more segments that are connected to another PE is referred to as a protector PE. A protector PE must learn the forwarding state of the segments that it is protecting from the primary PE that is being protected.

For a given segment, if the protector PE is not directly connected to the CE device associated with the segment, it must also learn the forwarding state from at least one backup PE. This situation might arise only in the case of egress PE failure protection.

A protector PE maintains forwarding state for a given segment in the context of the primary PE. A protector PE might maintain state for only a subset of the segments on the primary PE or for all the segments on the primary PE.

## Example: Configuring Egress Protection for Layer 3 VPN Services

This example shows how to configure egress protection for fast restoration of Layer 3 VPN services.

- [Requirements on page 276](#)
- [Overview on page 276](#)

- [Configuration on page 277](#)
- [Verification on page 282](#)

## Requirements

---

This example uses the following hardware and software components

- MX Series 3D Universal Edge Routers
- Tunnel PICs or the configuration of the Enhanced IP Network Services mode (using the **network-services enhanced-ip** statement at the **[edit chassis]** hierarchy level).
- Junos OS Release 11.4R3 or later running on the devices

Before you begin:

- Configure the device interfaces. See the *Junos OS Network Interfaces Configuration Guide*.
- Configure the following routing protocols on all the PE and PLR routers.
  - MPLS, LSPs, and LDP. See the *Junos OS MPLS Applications Configuration Guide*.
  - BGP and IS-IS. See the *Junos OS Routing Protocols Configuration Guide*.
- Configure Layer 3 VPNs. See the *Junos OS VPNs Configuration Guide*.

## Overview

---

Typically, Layer 3 VPN service restoration, in case of egress PE router failure (for multihomed customer edge [CE] routers), depends on the ingress PE router to detect the egress PE node failure and switch traffic to the backup PE router for multihomed CE sites.

Junos OS Release 11.4R3 or later enables you to configure egress protection for Layer 3 VPN services that protects the services from egress PE node failure in a scenario where the CE site is multihomed with more than one PE router. The mechanism enables local repair to be performed immediately upon an egress node failure. The router acting as the point of local repair (PLR) redirects VPN traffic to a protector PE router for restoring service quickly, achieving fast protection that is comparable to MPLS fast reroute.

The statements used to configure egress protection are:

- **egress-protection**—When configured at the **[edit protocols mpls]** hierarchy level, this statement specifies protector information and the context identifier for the Layer 3 VPN and edge protection virtual circuit:

```
[edit protocols mpls]
egress-protection {
  context-identifier context-id {
    primary | protector;
    metric igp-metric-value;
  }
}
```

When configured at the `[edit protocols bgp group group-name family inet-vpn unicast]`, `[edit protocols bgp group group-name family inet6-vpn unicast]`, or `[edit protocols bgp group group-name family iso-vpn unicast]` hierarchy levels, the `egress-protection` statement specifies the context identifier that enables egress protection for the configured BGP VPN network layer reachability information (NLRI).

```
[edit protocols bgp]
group internal {
  type internal;
  local-address ip-address;
  family <inet-vpn|inet6-vpn|iso-vpn> {
    unicast {
      egress-protection {
        context-identifier {
          context-id-ip-address;
        }
      }
    }
  }
}
```

When configured at the `[edit routing-instances]` hierarchy level, the `egress-protection` statement holds the context identifier of the protected PE router.

This configuration must be done only in the primary PE router and is used for outbound BGP updates for the next hops.

```
[edit routing-instance]
routing-instance-name {
  egress-protection {
    context-identifier {
      context-id-ip-address;
    }
  }
}
```

Configuring the `context-identifier` statement at the `[edit routing-instances routing-instance-name]` hierarchy level provides customer edge VRF-level context ID granularity for each VRF instance.

- **context-identifier**—This statement specifies an IPV4 address used to define the pair of PE routers participating in the egress protection LSP. The context identifier is used to assign an identifier to the protector PE router. The identifier is propagated to the other PE routers participating in the network, making it possible for the protected egress PE router to signal the egress protection LSP to the protector PE router.

## Configuration

### CLI Quick Configuration



**NOTE:** This example only shows sample configuration that is relevant to configuring egress PE protection for Layer 3 VPN services on the protected router, PE2, the protector router, PE3, and the PLR router.

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

PE2 (Protected PE Router)	<pre> set protocols mpls interface all set protocols mpls interface fxp0.0 disable set protocols mpls egress-protection context-identifier 66.6.6.6 primary set protocols bgp group ibgp type internal set protocols bgp group ibgp local-address 10.255.245.194 set protocols bgp group ibgp family inet-vpn unicast egress-protection context-identifier 66.6.6.6 </pre>
PE3 (Protector PE Router)	<pre> set protocols mpls interface all set protocols mpls interface fxp0.0 disable set protocols mpls egress-protection context-identifier 66.6.6.6 protector set protocols bgp group ibgp type internal set protocols bgp group ibgp local-address 10.255.245.196 set protocols bgp group ibgp family inet-vpn unicast egress-protection keep-import remote-vrf set policy-options policy-statement remote-vrf from community rsite1 set policy-options policy-statement remote-vrf from community rsite24 set policy-options policy-statement remote-vrf then accept set policy-options community rsite1 members target:1:1 set policy-options community rsite24 members target:100:1023 </pre>
PLR Router	<pre> set protocols mpls interface all set protocols mpls interface fxp0.0 disable set protocols isis level 1 disable set protocols isis interface all node-link-protection set protocols isis backup-spf-options per-prefix-calculation set protocols ldp track-igp-metric set protocols ldp interface all set protocols ldp interface fxp0.0 disable </pre>

#### *Configuring the Protected PE Router (PE2)*

##### **Step-by-Step Procedure**

To configure the protected PE router, PE2:

1. Configure MPLS on the interfaces.  

```

[edit protocols mpls]
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable

```
2. Configure egress protection and the context identifier.



**NOTE:** The context identifier type must be set to primary.

```

[edit protocols mpls]
user@PE2# set egress-protection context-identifier 66.6.6.6 primary

```

3. Configure egress protection for the configured BGP NRLI.



**NOTE:** The context identifier configured at the [edit protocols bgp group group-name family inet-vpn] hierarchy level should match the context identifier configured at the [edit protocols mpls] hierarchy level.

```
[edit protocols bgp]
user@PE2# set group ibgp type internal
user@PE2# set group ibgp local-address 10.255.245.194
user@PE2# set group ibgp family inet-vpn unicast egress-protection
context-identifier 66.6.6.6
```



**NOTE:** Configuring the context-identifier at the [edit routing-instances routing-instance-name] hierarchy level provides CE VRF-level context-id granularity for each virtual routing and forwarding (VRF) instance.

4. After you are done configuring the device, commit the configuration.

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

**Results** Confirm your configuration by issuing the show protocols command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show protocols
mpls {
  interface all;
  interface fxp0.0 {
    disable;
  }
  egress-protection {
    context-identifier 66.6.6.6 {
      primary;
    }
  }
}
bgp {
  group ibgp {
    type internal;
    local-address 10.255.245.194;
    family inet-vpn {
      unicast {
        egress-protection {
          context-identifier {
            66.6.6.6;
          }
        }
      }
    }
  }
}
```

```
}
```

### *Configuring the Protector PE Router (PE3)*

#### **Step-by-Step Procedure**

To configure the protector PE router, PE3:

1. Configure MPLS on the interfaces.  

```
[edit protocols mpls]
user@PE3# set interface all
user@PE3# set mpls interface fxp0.0 disable
```
2. Configure egress protection and the context identifier.  

```
[edit protocols mpls]
user@PE3# set egress-protection context-identifier 66.6.6.6 protector
```
3. Configure IPv4 Layer 3 VPN NRLI parameters.  

```
[edit protocols bgp]
user@PE3# set group ibgp type internal
user@PE3# set group ibgp local-address 10.255.245.196
user@PE3# set group ibgp family inet-vpn unicast egress-protection keep-import
remote-vrf
```
4. Configure routing policy options.  

```
[edit policy-options]
user@PE3# set policy-statement remote-vrf from community rsite1
user@PE3# set policy-statement remote-vrf from community rsite24
user@PE3# set policy-statement remote-vrf then accept
user@PE3# set community rsite1 members target:1:1
user@PE3# set community rsite24 members target:100:1023
```
5. After you are done configuring the device, commit the configuration.  

```
[edit]
user@PE3# commit
```

**Results** Confirm your configuration by issuing the **show protocols** and the **show policy-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show protocols
mpls {
  interface all;
  interface fxp0.0 {
    disable;
  }
  egress-protection {
    context-identifier 66.6.6.6 {
      protector;
    }
  }
}
bgp {
  group ibgp {
    type internal;
    local-address 10.255.245.196;
```

```

family inet-vpn {
  unicast {
    egress-protection {
      keep-import remote-vrf;
    }
  }
}
}

user@PE3# show policy-options
policy-statement remote-vrf {
  from community [ rsite1 rsite24 ];
  then accept;
}
community rsite1 members target:1:1;
community rsite24 members target:100:1023;

```

### Configuring the PLR Router

#### Step-by-Step Procedure

To configure the router acting as the point of local repair (PLR):

1. Configure MPLS on the interfaces.
 

```

[edit protocols mpls]
user@PLR# set interface all
user@PLR# set interface fxp0.0 disable
      
```
2. Configure per-prefix-LFA calculation along with link protection.
 

```

[edit protocols isis]
user@PLR# set backup-spf-options per-prefix-calculation
user@PLR# set level 1 disable
user@PLR# set interface all node-link-protection
user@PLR# set interface fxp0.0 disable
      
```
3. Configure LDP to use the interior gateway protocol (IGP) route metric instead of the default LDP route metric (the default LDP route metric is 1).
 

```

[edit protocols ldp]
user@PLR# set track-igp-metric
user@PLR# set interface all
user@PLR# set interface fxp0.0 disable
      
```

**Results** Confirm your configuration by issuing the **show protocols** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@PLR# show protocols
mpls {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
isis {
  backup-spf-options per-prefix-calculation;
}

```

```

    level 1 disable;
    interface all {
        node-link-protection;
    }
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}

```

### Verification

Confirm that the configuration is working properly.

- [Verifying Egress Protection Details on page 282](#)
- [Verifying Routing Instances on page 282](#)
- [Verifying BGP NRLI on page 283](#)

#### *Verifying Egress Protection Details*

**Purpose** Check the egress protection configuration.

**Action** user@PE3> show mpls egress-protection details

```

Instance          Type      Protection-Type
rsite1            remote-vrf  Protector
  RIB __66.6.6.6-rsite1__.inet.0, Context-Id 66.6.6.6, Enhanced-lookup
  Route Target 1:1
rsite24           remote-vrf  Protector
  RIB __66.6.6.6-rsite24__.inet.0, Context-Id 66.6.6.6, Enhanced-lookup
  Route Target 100:1023

```

**Meaning** **Instance** indicates the routing-instance name. **Type** shows the type of the VRF. It can be either **local-vrf** or **remote-vrf**. **RIB** (routing information base) indicates the edge-protection created routing table. **Context-Id** shows the context ID associated with the RIB. **Route Target** shows the route target associated with the routing instance.

#### *Verifying Routing Instances*

**Purpose** Verify the routing instances.

**Action** user@PE3> show route instance site1 detail

```
site1:
  Router ID: 1.5.0.1
  Type: vrf                      State: Active
  Interfaces:
    lt-1/3/0.8
  Route-distinguisher: 10.255.255.11:150
  Vrf-import: [ site1-import ]
  Vrf-export: [ __vrf-export-site1-internal__ ]
  Vrf-export-target: [ target:100:250 ]
  Fast-reroute-priority: low
  Vrf-edge-protection-id: 66.6.6.6
  Tables:
    site1.inet.0      : 27 routes (26 active, 0 holddown, 0 hidden)
    site1.iso.0       : 0 routes (0 active, 0 holddown, 0 hidden)
    site1.inet6.0     : 0 routes (0 active, 0 holddown, 0 hidden)
    site1.mdt.0       : 0 routes (0 active, 0 holddown, 0 hidden)
```

**Meaning** Vrf-edge-protection-id shows the egress protection configured in the protector PE router with the routing instance.

#### *Verifying BGP NLRI*

**Purpose** Check the details of the BGP VPN network layer reachability information.

**Action** user@PE3> show bgp neighbor

```
Peer: 10.255.55.1+179 AS 65535 Local: 10.255.22.1+59264 AS 65535
  Type: Internal  State: Established  Flags: <ImportEval Sync>
  Last State: OpenConfirm  Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress KeepAll AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 10.255.22.1 Holdtime: 90 Preference: 170
  NLRI configured with egress-protection: inet-vpn-unicast
  Egress-protection NLRI inet-vpn-unicast, keep-import: [ VPN-A-remote ]
  Number of flaps: 0
```

**Meaning** NLRI configured with egress-protection shows the BGP family configured with egress protection. egress-protection NLRI inet-vpn-unicast, keep-import: [remote-vrf] shows the egress protection routing policy for the BGP group.

## Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP

This example shows how to configure fast service restoration at the egress of a Layer 3 VPN when the customer is multihomed to the service provider. Further, this example includes enhanced point-of-local-repair (PLR) functionality, in which the PLR reroutes service traffic during an egress failure.

Starting in Junos OS Release 13.3, enhanced PLR functionality is available, in which the PLR reroutes service traffic during an egress failure. As part of this enhancement, the PLR router no longer needs to be directly connected to the protector router. Previously,

if the PLR was not directly connected to the protector router, the loop-free alternate route could not find the backup path to the protector.

- [Requirements on page 284](#)
- [Overview on page 284](#)
- [Configuration on page 285](#)
- [Verification on page 299](#)

## Requirements

No special configuration beyond device initialization is required before configuring this example.

This example requires Junos OS Release 13.3 or later.

## Overview

In this example, the customer edge (CE) devices are part of a VPN where Device CE1 is multihomed with Device PE2 and Device PE3.

Device PE3 acts as the protector for the Layer 3 VPN routing instances or subnets.

Device PE1 is the originator for the context identifier for Device CE1, Device PE2 is the primary router for that context identifier, while Device PE3 is the protector for that context identifier.

Device P1 acts as the point of local repair (PLR). As such, Device P1 can redirect Layer 3 VPN traffic to the protector PE router to enable fast restoration and reroute.

The working path is through P1>PE2. The backup path is through P1>PE3. Traffic flows through the working path under normal circumstances. When a Device PE2 node or link failure is detected, traffic is rerouted from the working path to the protected path. In the normal failover process, the detection of failure and the recovery rely on the control plane and is therefore relatively slow. Typically, if there is a link or node failure in the core network, the egress PE router would have to rely on the ingress PE router to detect the failure and switch over to the backup path, because a local repair option for egress failure is not available. To provide a local repair solution for the egress PE link or node failure, a mechanism known as egress protection is used in this example to repair and restore the connection quickly. Because egress protection is configured, the PLR router detects the Device PE2 link or node failure and reroutes traffic through the protector Device PE3 using the backup LDP-signaled label-switched path (LSP). The PLR router uses per-prefix loop-free alternate routes to program the backup next hop through Device PE3, and traffic is forwarded to Device CE2 using the alternate paths. This restoration is done quickly after the PLR router detects the Device PE2 egress node or link failure. The dual protection mechanism can also be used for egress protection where the two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops.

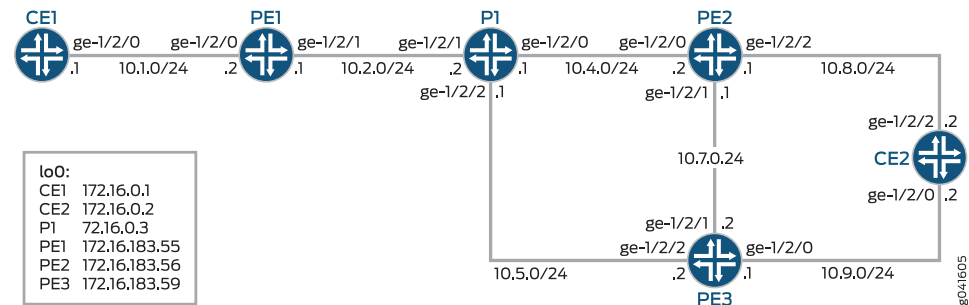
In addition to egress protection, this example demonstrates an enhanced PLR function, in which the PLR reroutes service traffic during the egress failure. This enhancement is supported in Junos OS Release 13.3 and later. In this example, Device P1 (the PLR) is

directly connected to Device PE3 (the protector). A new configuration statement, **advertise-mode**, enables you to set the method for the interior gateway protocol (IGP) to advertise egress protection availability.

### Topology

Figure 38 on page 285 shows the sample network.

Figure 38: Layer 3 VPN Egress Protection with RSVP and LDP



### Configuration

- CLI Quick Configuration** To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.
- Device CE1**
- ```
set interfaces ge-1/2/0 unit 0 description to_PE1
set interfaces ge-1/2/0 unit 0 family inet address 10.1.0.1/24
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```
- Device CE2**
- ```
set interfaces ge-1/2/2 unit 0 description to_PE2
set interfaces ge-1/2/2 unit 0 family inet address 10.8.0.2/24
set interfaces ge-1/2/0 unit 0 description to_PE3
set interfaces ge-1/2/0 unit 0 family inet address 10.9.0.2/24
set interfaces lo0 unit 0 family inet address 172.16.0.2/32
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```
- Device P1**
- ```
set interfaces ge-1/2/1 unit 0 description to_PE1
set interfaces ge-1/2/1 unit 0 family inet address 10.2.0.2/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/0 unit 0 description to_PE2
set interfaces ge-1/2/0 unit 0 family inet address 10.4.0.1/24
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 description to_PE3
set interfaces ge-1/2/2 unit 0 family inet address 10.5.0.1/24
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family iso address 49.0002.0172.0016.0003.00
set protocols rsvp interface all
```

```

set protocols rsvp interface fxp0.0 disable
set protocols mpls interface all
set protocols isis backup-spf-options per-prefix-calculation
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all node-link-protection
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

```

```

Device PE1
set interfaces ge-1/2/0 unit 0 description to_CE1
set interfaces ge-1/2/0 unit 0 family inet address 10.1.0.2/24
set interfaces ge-1/2/1 unit 0 description to_P1
set interfaces ge-1/2/1 unit 0 family inet address 10.2.0.1/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.55/32
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3055.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPrimary6.6.6.6 to 6.6.6.6
set protocols mpls label-switched-path toPrimary6.6.6.6 egress-protection
set protocols mpls interface all
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.55
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 172.16.183.56
set protocols bgp group ibgp neighbor 172.16.183.59
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-1/2/0.0
set routing-instances vpn1 route-distinguisher 172.16.183.55:10
set routing-instances vpn1 vrf-target target:10:10
set routing-instances vpn1 routing-options static route 100.0.0.0/24 next-hop 10.1.0.1
set routing-instances vpn1 protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set routing-options autonomous-system 64510

```

```

Device PE2
set interfaces ge-1/2/0 unit 0 description to_P1
set interfaces ge-1/2/0 unit 0 family inet address 10.4.0.2/24
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 description to_CE2
set interfaces ge-1/2/2 unit 0 family inet address 10.8.0.1/24
set interfaces ge-1/2/1 unit 0 description to_PE3
set interfaces ge-1/2/1 unit 0 family inet address 10.7.0.1/24
set interfaces ge-1/2/1 unit 0 family iso

```

```

set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.56/32
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3056.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPE1 to 172.16.183.55
set protocols mpls label-switched-path toPrimary6.6.6 to 6.6.6.6
set protocols mpls label-switched-path toPrimary6.6.6 egress-protection
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 6.6.6.6 primary
set protocols mpls egress-protection context-identifier 6.6.6.6 advertise-mode stub-proxy
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.56
set protocols bgp group ibgp family inet-vpn unicast egress-protection context-identifier
    6.6.6.6
set protocols bgp group ibgp neighbor 172.16.183.55
set protocols bgp group ibgp neighbor 172.16.183.59
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 64510

```

**Device PE3**

```

set interfaces ge-1/2/2 unit 0 description to_P1
set interfaces ge-1/2/2 unit 0 family inet address 10.5.0.2/24
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/0 unit 0 description to_CE2
set interfaces ge-1/2/0 unit 0 family inet address 10.9.0.1/24
set interfaces ge-1/2/1 unit 0 description to_PE2
set interfaces ge-1/2/1 unit 0 family inet address 10.7.0.2/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.59/32
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3059.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPE1 to 172.16.183.55
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 6.6.6.6 protector
set protocols mpls egress-protection context-identifier 6.6.6.6 advertise-mode stub-proxy
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.59
set protocols bgp group ibgp family inet-vpn unicast egress-protection keep-import
    remote-vrf
set protocols bgp group ibgp neighbor 172.16.183.55
set protocols bgp group ibgp neighbor 172.16.183.56
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0

```

```

set protocols ldp track-igp-metric
set protocols ldp interface all
set policy-options policy-statement remote-vrf from community rsite1
set policy-options policy-statement remote-vrf from community rsite24
set policy-options policy-statement remote-vrf then accept
set policy-options community rsite1 members target:1:1
set policy-options community rsite24 members target:100:1023
set routing-options autonomous-system 64510

```

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

To configure Device P1 (the PLR):

1. Configure the device interfaces.

```

[edit interfaces]
user@P1# set ge-1/2/1 unit 0 description to_PE1
user@P1# set ge-1/2/1 unit 0 family inet address 10.2.0.2/24
user@P1# set ge-1/2/1 unit 0 family iso
user@P1# set ge-1/2/1 unit 0 family mpls

user@P1# set ge-1/2/0 unit 0 description to_PE2
user@P1# set ge-1/2/0 unit 0 family inet address 10.4.0.1/24
user@P1# set ge-1/2/0 unit 0 family iso
user@P1# set ge-1/2/0 unit 0 family mpls

user@P1# set ge-1/2/2 unit 0 description to_PE3
user@P1# set ge-1/2/2 unit 0 family inet address 10.5.0.1/24
user@P1# set ge-1/2/2 unit 0 family iso
user@P1# set ge-1/2/2 unit 0 family mpls

user@P1# set lo0 unit 0 family inet address 172.16.0.3/32
user@P1# set lo0 unit 0 family iso address 49.0002.0172.0016.0003.00

```

2. Configure IS-IS.

Configure per-prefix-LFA calculation along with node link protection.

```

[edit protocols isis]
user@P1# set backup-spf-options per-prefix-calculation
user@P1# set level 1 disable
user@P1# set level 2 wide-metrics-only
user@P1# set interface all node-link-protection
user@P1# set interface fxp0.0 disable
user@P1# set interface lo0.0

```

3. Enable MPLS.

```

[edit protocols mpls ]
user@P1# set interface all

```

4. Enable RSVP.

```

[edit protocols rsvp]
user@P1# set interface all

```

```
user@P1# set interface fxp0.0 disable
```

5. Enable LDP.

```
[edit protocols ldp]
user@P1# set track-igp-metric
user@P1# set interface all
user@P1# set interface fxp0.0 disable
```

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

To configure Device PE1:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE1# set ge-1/2/0 unit 0 description to_CE1
user@PE1# set ge-1/2/0 unit 0 family inet address 10.1.0.2/24
```

```
user@PE1# set ge-1/2/1 unit 0 description to_P1
user@PE1# set ge-1/2/1 unit 0 family inet address 10.2.0.1/24
user@PE1# set ge-1/2/1 unit 0 family iso
user@PE1# set ge-1/2/1 unit 0 family mpls
```

```
user@PE1# set lo0 unit 0 family inet address 172.16.183.55/32
user@PE1# set lo0 unit 0 family iso address 49.0002.1720.1618.3055.00
```

2. Enable RSVP.

```
[edit protocols rsvp]
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
```

3. Configure MPLS.

```
[edit protocols mpls]
user@PE1# set label-switched-path toPrimary6.6.6.6 to 6.6.6.6
user@PE1# set label-switched-path toPrimary6.6.6.6 egress-protection
user@PE1# set interface all
```

4. Configure IBGP.

```
[edit protocols bgp group ibgp]
user@PE1# set type internal
user@PE1# set local-address 172.16.183.55
user@PE1# set family inet-vpn unicast
user@PE1# set neighbor 172.16.183.56
user@PE1# set neighbor 172.16.183.59
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE1# set level 1 disable
user@PE1# set level 2 wide-metrics-only
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
```

```
user@PE1# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE1# set track-igp-metric
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
```

7. Configure the routing instance.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 172.16.183.55:10
user@PE1# set vrf-target target:10:10
user@PE1# set routing-options static route 100.0.0.0/24 next-hop 10.1.0.1
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```

8. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@PE1# set autonomous-system 64510
```

#### Step-by-Step Procedure

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

To configure Device PE2:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE2# set ge-1/2/0 unit 0 description to_P1
user@PE2# set ge-1/2/0 unit 0 family inet address 10.4.0.2/24
user@PE2# set ge-1/2/0 unit 0 family iso
user@PE2# set ge-1/2/0 unit 0 family mpls
```

```
user@PE2# set ge-1/2/2 unit 0 description to_CE2
user@PE2# set ge-1/2/2 unit 0 family inet address 10.8.0.1/24
```

```
user@PE2# set ge-1/2/1 unit 0 description to_PE3
user@PE2# set ge-1/2/1 unit 0 family inet address 10.7.0.1/24
user@PE2# set ge-1/2/1 unit 0 family iso
user@PE2# set ge-1/2/1 unit 0 family mpls
```

```
user@PE2# set lo0 unit 0 family inet address 172.16.183.56/32
user@PE2# set lo0 unit 0 family iso address 49.0002.1720.1618.3056.00
```

2. Enable RSVP.

```
[edit protocols rsvp]
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
```

3. Configure MPLS.

```
[edit protocols mpls]
user@PE2# set label-switched-path toPE1 to 172.16.183.55
user@PE2# set label-switched-path toPrimary6.6.6.6 to 6.6.6.6
user@PE2# set label-switched-path toPrimary6.6.6.6 egress-protection
user@PE2# set interface all
user@PE2# set egress-protection context-identifier 6.6.6.6 primary
user@PE2# set egress-protection context-identifier 6.6.6.6 advertise-mode
stub-proxy
```

4. Configure IBGP.

```
[edit protocols bgp group ibgp]
user@PE2# set type internal
user@PE2# set local-address 172.16.183.56
user@PE2# set family inet-vpn unicast egress-protection context-identifier 6.6.6.6
user@PE2# set neighbor 172.16.183.55
user@PE2# set neighbor 172.16.183.59
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE2# set level 1 disable
user@PE2# set level 2 wide-metrics-only
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
user@PE2# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE2# set track-igp-metric
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
```

7. Configure the AS number.

```
[edit routing-options]
user@PE2# set autonomous-system 64510
```

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

To configure Device PE3:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE3# set ge-1/2/2 unit 0 description to_P1
user@PE3# set ge-1/2/2 unit 0 family inet address 10.5.0.2/24
user@PE3# set ge-1/2/2 unit 0 family iso
user@PE3# set ge-1/2/2 unit 0 family mpls

user@PE3# set ge-1/2/0 unit 0 description to_CE2
user@PE3# set ge-1/2/0 unit 0 family inet address 10.9.0.1/24

user@PE3# set ge-1/2/1 unit 0 description to_PE2
```

```
user@PE3# set ge-1/2/1 unit 0 family inet address 10.7.0.2/24
user@PE3# set ge-1/2/1 unit 0 family iso
user@PE3# set ge-1/2/1 unit 0 family mpls
```

```
user@PE3# set lo0 unit 0 family inet address 172.16.183.59/32
user@PE3# set lo0 unit 0 family iso address 49.0002.1720.1618.3059.00
```

2. Enable RSVP.

```
[edit protocols rsvp]
user@PE3# set interface all
user@PE3# set interface fxp0.0 disable
```

3. Configure MPLS.

```
[edit protocols mpls]
user@PE3# set label-switched-path toPE1 to 172.16.183.55
user@PE3# set interface all
user@PE3# set egress-protection context-identifier 6.6.6.6 protector
user@PE3# set egress-protection context-identifier 6.6.6.6 advertise-mode
stub-proxy
```

4. Configure IBGP.

```
[edit protocols bgp group ibgp]
user@PE3# set type internal
user@PE3# set local-address 172.16.183.59
user@PE3# set family inet-vpn unicast egress-protection keep-import remote-vrf
user@PE3# set neighbor 172.16.183.55
user@PE3# set neighbor 172.16.183.56
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE3# set level 1 disable
user@PE3# set level 2 wide-metrics-only
user@PE3# set interface all
user@PE3# set interface fxp0.0 disable
user@PE3# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE3# set track-igp-metric
user@PE3# set interface all
```

7. Configure the routing policy.

```
[edit policy-options]
user@PE3# set policy-statement remote-vrf from community rsite1
user@PE3# set policy-statement remote-vrf from community rsite24
user@PE3# set policy-statement remote-vrf then accept
user@PE3# set community rsite1 members target:1:1
user@PE3# set community rsite24 members target:100:1023
```

8. Configure the AS number.

```
[edit routing-options]
user@PE3# set autonomous-system 64510
```

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

```

Device P1 user@P1# show interfaces
ge-1/2/0 {
  unit 0 {
    description to_PE2;
    family inet {
      address 10.4.0.1/24;
    }
    family iso;
    family mpls;
  }
}
ge-1/2/1 {
  unit 0 {
    description to_PE1;
    family inet {
      address 10.2.0.2/24;
    }
    family iso;
    family mpls;
  }
}
ge-1/2/2 {
  unit 0 {
    description to_PE3;
    family inet {
      address 10.5.0.1/24;
    }
    family iso;
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 172.16.0.3/32;
    }
    family iso {
      address 49.0002.0172.0016.0003.00;
    }
  }
}

user@P1# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
mpls {
  interface all;

```

```
}
isis {
  backup-spf-options per-prefix-calculation;
  level 1 disable;
  level 2 wide-metrics-only;
  interface all {
    node-link-protection;
  }
  interface fxp0.0 {
    disable;
  }
  interface lo0.0;
}
ldp {
  track-igp-metric;
  interface all;
  interface fxp0.0 {
    disable;
  }
}
```

**Device PE1**

```
user@PE1# show interfaces
ge-1/2/0 {
  unit 0 {
    description to_CE1;
    family inet {
      address 10.1.0.2/24;
    }
  }
}
ge-1/2/1 {
  unit 0 {
    description to_P1;
    family inet {
      address 10.2.0.1/24;
    }
    family iso;
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 172.16.183.55/32;
    }
    family iso {
      address 49.0002.1720.1618.3055.00;
    }
  }
}

user@PE1# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
```

```

        disable;
    }
}
mpls {
    label-switched-path toPE2Primary6.6.6.6 {
        to 6.6.6.6;
        egress-protection;
    }
    interface all;
}
bgp {
    group ibgp {
        type internal;
        local-address 172.16.183.55;
        family inet-vpn {
            unicast;
        }
        neighbor 172.16.183.56;
        neighbor 172.16.183.59;
    }
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}

user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 172.16.183.55:10;
    vrf-target target:10:10;
    routing-options {
        static {
            route 100.0.0.0/24 next-hop 10.1.0.1;
        }
    }
    protocols {
        ospf {
            area 0.0.0.0 {
                interface ge-1/2/0.0;
            }
        }
    }
}

```

```

}
user@PE1# show routing-options
autonomous-system 64510;

```

**Device PE2**

```

user@PE2# show interfaces
ge-1/2/0 {
  unit 0 {
    description to_P1;
    family inet {
      address 10.4.0.2/24;
    }
    family iso;
    family mpls;
  }
}
ge-1/2/1 {
  unit 0 {
    description to_PE3;
    family inet {
      address 10.7.0.1/24;
    }
    family iso;
    family mpls;
  }
}
ge-1/2/2 {
  unit 0 {
    description to_CE2;
    family inet {
      address 10.8.0.1/24;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 172.16.183.56/32;
    }
    family iso {
      address 49.0002.1720.1618.3056.00;
    }
  }
}

user@PE2# show protocols
rsvp {
  interface all;
  interface fxp0.0 {
    disable;
  }
}
mpls {
  label-switched-path toPE1 {
    to 172.16.183.55;
  }
}

```

```

    }
    label-switched-path toPE2Primary6.6.6.6 {
        to 6.6.6.6;
        egress-protection;
    }
    interface all;
    egress-protection {
        context-identifier 6.6.6.6 {
            primary;
            advertise-mode stub-proxy;
        }
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 172.16.183.56;
        family inet-vpn {
            unicast {
                egress-protection {
                    context-identifier {
                        6.6.6.6;
                    }
                }
            }
        }
    }
    neighbor 172.16.183.55;
    neighbor 172.16.183.59;
}
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}

```

```

user@PE2# show routing-options
autonomous-system 64510;

```

#### Device PE3

```

user@PE3# show interfaces
ge-1/2/0 {
    unit 0 {
        description to_CE2;
        family inet {

```

```
        address 10.9.0.1/24;
    }
}
ge-1/2/1 {
    unit 0 {
        description to_PE2;
        family inet {
            address 10.7.0.2/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 0 {
        description to_P1;
        family inet {
            address 10.5.0.2/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.183.59/32;
        }
        family iso {
            address 49.0002.1720.1618.3059.00;
        }
    }
}

user@PE3# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path toPE1 {
        to 172.16.183.55;
    }
    interface all;
    egress-protection {
        context-identifier 6.6.6.6 {
            protector;
            advertise-mode stub-proxy;
        }
    }
}
bgp {
    group ibgp {
```

```

    type internal;
    local-address 172.16.183.59;
    family inet-vpn {
        unicast {
            egress-protection {
                keep-import remote-vrf;
            }
        }
    }
    neighbor 172.16.183.55;
    neighbor 172.16.183.56;
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
}

user@PE3# show policy-options
policy-statement remote-vrf {
    from community [ rsite1 rsite24 ];
    then accept;
}
community rsite1 members target:1:1;
community rsite24 members target:100:1023;

user@PE3# show routing-options
autonomous-system 64510;

```

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

## Verification

Confirm that the configuration is working properly.

- [Verifying the Protector Node on page 300](#)
- [Verifying the Primary Node on page 300](#)
- [Checking the Context Identifier Route on page 300](#)
- [Verifying Egress Protection on page 301](#)
- [Verifying the Routing Instance on Device PE1 on page 302](#)
- [Verifying the LSPs on page 302](#)
- [Verifying BGP NRLI on page 307](#)

- [Verifying the Traffic Engineering Database on page 309](#)
- [Verifying the IS-IS Database on page 314](#)

---

### Verifying the Protector Node

---

- Purpose** On the protector node (Device PE3), check the information about configured egress protection context identifiers.
- Action** user@PE3> `show mpls context-identifier detail protector`
- ```
ID: 6.6.6.6
Type: protector, Metric: 16777215, Mode: proxy
Context table: __PE3:6.6.6.6__.mpls.0
Context LSPs:
  toPE2Primary6.6.6.6, from: 172.16.183.55
  toPE2Primary6.6.6.6, from: 172.16.183.56

Total 1, Primary 0, Protector 1
```
- Meaning** Device PE3 is the protector node for two LSPs configured from Device PE1 (172.16.183.55) and Device PE2 (172.16.183.56).

---

### Verifying the Primary Node

---

- Purpose** On the primary node (Device PE2), check the information about configured egress protection context identifiers.
- Action** user@PE2> `show mpls context-identifier detail primary`
- ```
ID: 6.6.6.6
Type: primary, Metric: 1, Mode: proxy

Total 1, Primary 1, Protector 0
```
- Meaning** Device PE2 is the primary node.

---

### Checking the Context Identifier Route

---

- Purpose** Examine the information about the context identifier (6.6.6.6).

**Action** user@PE1> show route 6.6.6.6

```
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[IS-IS/18] 00:53:39, metric 21
                    > to 10.2.0.2 via ge-1/2/1.0
```

```
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[LDP/9] 00:53:39, metric 21
                    > to 10.2.0.2 via ge-1/2/1.0, Push 299808
```

user@PE2> show route 6.6.6.6

```
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[MPLS/1] 3d 02:53:37, metric 1
                    Receive
                    [IS-IS/18] 00:06:08, metric 16777224
                    > to 10.7.0.2 via ge-1/2/1.0
```

user@PE3> show route 6.6.6.6

```
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[MPLS/2] 3d 02:53:36, metric 16777215
                    Receive
                    [IS-IS/18] 3d 02:53:28, metric 11
                    > to 10.7.0.1 via ge-1/2/1.0
```

user@P1> show route 6.6.6.6

```
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[IS-IS/18] 00:53:40, metric 11
                    > to 10.4.0.2 via ge-1/2/0.0
```

```
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
6.6.6.6/32          *[LDP/9] 00:53:40, metric 11
                    > to 10.4.0.2 via ge-1/2/0.0
```

### Verifying Egress Protection

**Purpose** On Device PE3, check the routes in the routing table.

**Action** user@PE3> show mpls egress-protection detail

| Instance              | Type       | Protection-Type |
|-----------------------|------------|-----------------|
| rsite1                | remote-vrf | Protector       |
| Route Target 1:1      |            |                 |
| rsite24               | remote-vrf | Protector       |
| Route Target 100:1023 |            |                 |

**Meaning** **Instance** indicates the community name. **Type** shows the type of the VRF. It can be either **local-vrf** or **remote-vrf**. **Route Target** shows the route target associated with the routing instance.

---

### Verifying the Routing Instance on Device PE1

**Purpose** On Device PE1, check the routes in the routing table.

**Action** user@PE1> show route instance vpn1 detail

vpn1:

Router ID: 10.1.0.2

Type: vrf State: Active

Interfaces:

ge-1/2/0.0

Route-distinguisher: 172.16.183.55:10

Vrf-import: [ \_\_vrf-import-vpn1-internal\_\_ ]

Vrf-export: [ \_\_vrf-export-vpn1-internal\_\_ ]

Vrf-import-target: [ target:10:10 ]

Vrf-export-target: [ target:10:10 ]

Fast-reroute-priority: low

Tables:

vpn1.inet.0 : 4 routes (4 active, 0 holddown, 0 hidden)

---

### Verifying the LSPs

**Purpose** On all devices, check the LSP information.

**Action** user@PE1> show mpls lsp extensive

Ingress LSP: 1 sessions

6.6.6.6

```

From: 172.16.183.55, State: Up, ActiveRoute: 0, LSPname: toPE2Primary6.6.6.6
ActivePath: (primary)
LSPath: Static Configured, Penultimate hop popping
LoadBalance: Random
Encoding type: Packet, Switching type: Packet, GPID: IPv4
*Primary State: Up
Priorities: 7 0
SmartOptimizeTimer: 180
Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 16777234)
10.2.0.2 S 10.5.0.2 S 6.6.6.6 S (link-id=2)
Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt
20=Node-ID):
10.2.0.2 10.5.0.2
17 Jun 10 13:13:04.973 CSPF: computation result accepted 10.2.0.2 10.5.0.2
6.6.6.6(link-id=2)
16 Jun 10 13:12:36.155 CSPF failed: no route toward 6.6.6.6[4 times]
15 Jun 10 13:11:26.269 CSPF: link down/deleted:
0.0.0.0(172.16.183.59:2147618818)(PE3.00/172.16.183.59)->0.0.0.0(6.6.6.6:2)(PE2-6.6.6.6.00/6.6.6.6)

14 Jun 10 13:10:11.771 Selected as active path
13 Jun 10 13:10:11.770 Record Route: 10.2.0.2 10.5.0.2
12 Jun 10 13:10:11.770 Up
11 Jun 10 13:10:11.634 Originate Call
10 Jun 10 13:10:11.634 CSPF: computation result accepted 10.2.0.2 10.5.0.2
6.6.6.6(link-id=2)
9 Jun 10 13:10:11.623 Clear Call
8 Jun 10 13:10:11.622 Deselected as active
7 Jun 7 11:23:08.224 Selected as active path
6 Jun 7 11:23:08.224 Record Route: 10.2.0.2 10.5.0.2
5 Jun 7 11:23:08.223 Up
4 Jun 7 11:23:08.116 Originate Call
3 Jun 7 11:23:08.116 CSPF: computation result accepted 10.2.0.2 10.5.0.2
6.6.6.6(link-id=2)
2 Jun 7 11:22:38.132 CSPF failed: no route toward 6.6.6.6
1 Jun 7 11:22:08.607 CSPF: could not determine self[8 times]
Created: Fri Jun 7 11:18:46 2013
Total 1 displayed, Up 1, Down 0

```

Egress LSP: 2 sessions

172.16.183.55

```

From: 172.16.183.59, LSPstate: Up, ActiveRoute: 0
LSPname: toPE1, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: -
Resv style: 1 FF, Label in: 3, Label out: -
Time left: 126, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 10941 protocol 0
PATH rcvfrom: 10.2.0.2 (ge-1/2/1.0) 105 pkts
Adspec: received MTU 1500
PATH sentto: localclient
RESV rcvfrom: localclient
Record route: 10.5.0.2 10.2.0.2 <self>

```

```

172.16.183.55
  From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE1, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: -
  Resv style: 1 FF, Label in: 3, Label out: -
  Time left: 156, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 59956 protocol 0
  PATH rcvfrom: 10.2.0.2 (ge-1/2/1.0) 105 pkts
  Adspec: received MTU 1500
  PATH sentto: localclient
  RESV rcvfrom: localclient
  Record route: 10.4.0.2 10.2.0.2 <self>
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
-----

user@PE2> show mpls lsp extensive
Ingress LSP: 2 sessions

6.6.6.6
  From: 172.16.183.56, State: Up, ActiveRoute: 0, LSPname: toPE2Primary6.6.6.6
  ActivePath: (primary)
  LSPtype: Static Configured, Penultimate hop popping
  LoadBalance: Random
  Encoding type: Packet, Switching type: Packet, GPID: IPv4
  *Primary State: Up
    Priorities: 7 0
    SmartOptimizeTimer: 180
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 16777224)
10.7.0.2 S 6.6.6.6 S (link-id=2)
  Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt
20=Node-ID):
    10.7.0.2
      16 Jun 10 13:13:07.220 CSPF: computation result accepted 10.7.0.2
6.6.6.6(link-id=2)
      15 Jun 10 13:12:38.250 CSPF failed: no route toward 6.6.6.6[4 times]
      14 Jun 10 13:11:26.258 CSPF: link down/deleted:
0.0.0.0(172.16.183.59:2147618818)(PE3.00/172.16.183.59)->0.0.0.0(6.6.6.6:2)(PE2-6.6.6.6.00/6.6.6.6)

      13 Jun 10 13:10:11.746 Selected as active path
      12 Jun 10 13:10:11.743 Record Route: 10.7.0.2
      11 Jun 10 13:10:11.742 Up
      10 Jun 10 13:10:11.680 Originate Call
      9 Jun 10 13:10:11.680 CSPF: computation result accepted 10.7.0.2
6.6.6.6(link-id=2)
      8 Jun 10 13:10:11.674 Clear Call
      7 Jun 10 13:10:11.669 Deselected as active
      6 Jun 7 11:23:09.370 Selected as active path
      5 Jun 7 11:23:09.370 Record Route: 10.7.0.2
      4 Jun 7 11:23:09.369 Up
      3 Jun 7 11:23:09.349 Originate Call
      2 Jun 7 11:23:09.349 CSPF: computation result accepted 10.7.0.2
6.6.6.6(link-id=2)
      1 Jun 7 11:22:40.140 CSPF failed: no route toward 6.6.6.6[9 times]
      Created: Fri Jun 7 11:18:46 2013

172.16.183.55
  From: 172.16.183.56, State: Up, ActiveRoute: 0, LSPname: toPE1

```

```

ActivePath: (primary)
LSPTYPE: Static Configured, Penultimate hop popping
LoadBalance: Random
Encoding type: Packet, Switching type: Packet, GPID: IPv4
*Primary                               State: Up
  Priorities: 7 0
  SmartOptimizeTimer: 180
  Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 20)
10.4.0.1 S 10.2.0.1 S
  Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt
20=Node-ID):
    10.4.0.1 10.2.0.1
13 Jun 10 13:10:11.794 Selected as active path
12 Jun 10 13:10:11.793 Record Route: 10.4.0.1 10.2.0.1
11 Jun 10 13:10:11.793 Up
10 Jun 10 13:10:11.679 Originate Call
9 Jun 10 13:10:11.679 CSPF: computation result accepted 10.4.0.1 10.2.0.1
8 Jun 10 13:10:11.660 Clear Call
7 Jun 10 13:10:11.645 Deselected as active
6 Jun 7 11:22:40.031 Selected as active path
5 Jun 7 11:22:40.024 Record Route: 10.4.0.1 10.2.0.1
4 Jun 7 11:22:40.012 Up
3 Jun 7 11:22:39.687 Originate Call
2 Jun 7 11:22:39.687 CSPF: computation result accepted 10.4.0.1 10.2.0.1
1 Jun 7 11:22:10.235 CSPF failed: no route toward 172.16.183.55[8 times]
Created: Fri Jun 7 11:18:45 2013
Total 2 displayed, Up 2, Down 0

Egress LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

user@PE3> show mpls lsp extensive
Ingress LSP: 1 sessions

172.16.183.55
From: 172.16.183.59, State: Up, ActiveRoute: 0, LSPname: toPE1
ActivePath: (primary)
LSPTYPE: Static Configured, Penultimate hop popping
LoadBalance: Random
Encoding type: Packet, Switching type: Packet, GPID: IPv4
*Primary                               State: Up
  Priorities: 7 0
  SmartOptimizeTimer: 180
  Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 20)
10.5.0.1 S 10.2.0.1 S
  Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt
20=Node-ID):
    10.5.0.1 10.2.0.1
13 Jun 10 13:10:11.708 Selected as active path
12 Jun 10 13:10:11.703 Record Route: 10.5.0.1 10.2.0.1
11 Jun 10 13:10:11.703 Up
10 Jun 10 13:10:11.599 Originate Call
9 Jun 10 13:10:11.599 CSPF: computation result accepted 10.5.0.1 10.2.0.1
8 Jun 10 13:10:11.558 Clear Call
7 Jun 10 13:10:11.555 Deselected as active
6 Jun 7 11:22:41.829 Selected as active path
5 Jun 7 11:22:41.828 Record Route: 10.5.0.1 10.2.0.1
4 Jun 7 11:22:41.827 Up
3 Jun 7 11:22:41.767 Originate Call

```

```

    2 Jun  7 11:22:41.767 CSPF: computation result accepted 10.5.0.1 10.2.0.1
    1 Jun  7 11:22:12.289 CSPF failed: no route toward 172.16.183.55[8 times]
Created: Fri Jun  7 11:18:45 2013
Total 1 displayed, Up 1, Down 0

```

Egress LSP: 2 sessions

6.6.6.6

```

From: 172.16.183.55, LSPstate: Up, ActiveRoute: 0
LSPname: toPE2Primary6.6.6.6, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: -
Resv style: 1 FF, Label in: 299920, Label out: 3
Time left: 141, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 17060 protocol 0
Attrib flags: Non-PHP OOB
PATH rcvfrom: 10.5.0.1 (ge-1/2/2.0) 105 pkts
Adspec: received MTU 1500
PATH sentto: localclient
RESV rcvfrom: localclient
Record route: 10.2.0.1 10.5.0.1 <self>

```

6.6.6.6

```

From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
LSPname: toPE2Primary6.6.6.6, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: -
Resv style: 1 FF, Label in: 299936, Label out: 3
Time left: 152, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 59957 protocol 0
Attrib flags: Non-PHP OOB
PATH rcvfrom: 10.7.0.1 (ge-1/2/1.0) 106 pkts
Adspec: received MTU 1500
PATH sentto: localclient
RESV rcvfrom: localclient
Record route: 10.7.0.1 <self>

```

Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

user@P1> show mpls lsp extensive

Ingress LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

Egress LSP: 0 sessions

Total 0 displayed, Up 0, Down 0

Transit LSP: 3 sessions

6.6.6.6

```

From: 172.16.183.55, LSPstate: Up, ActiveRoute: 0
LSPname: toPE2Primary6.6.6.6, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: 299920
Resv style: 1 FF, Label in: 299904, Label out: 299920
Time left: 141, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 17060 protocol 0
Attrib flags: Non-PHP OOB

```

```

PATH rcvfrom: 10.2.0.1 (ge-1/2/1.0) 106 pkts
Adspec: received MTU 1500 sent MTU 1500
PATH sentto: 10.5.0.2 (ge-1/2/2.0) 105 pkts
RESV rcvfrom: 10.5.0.2 (ge-1/2/2.0) 105 pkts
Explct route: 10.5.0.2 6.6.6.6 (link-id=2)
Record route: 10.2.0.1 <self> 10.5.0.2

172.16.183.55
From: 172.16.183.59, LSPstate: Up, ActiveRoute: 0
LSPname: toPE1, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: 3
Resv style: 1 FF, Label in: 299888, Label out: 3
Time left: 158, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 10941 protocol 0
PATH rcvfrom: 10.5.0.2 (ge-1/2/2.0) 106 pkts
Adspec: received MTU 1500 sent MTU 1500
PATH sentto: 10.2.0.1 (ge-1/2/1.0) 105 pkts
RESV rcvfrom: 10.2.0.1 (ge-1/2/1.0) 105 pkts
Explct route: 10.2.0.1
Record route: 10.5.0.2 <self> 10.2.0.1

172.16.183.55
From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
LSPname: toPE1, LSPpath: Primary
Suggested label received: -, Suggested label sent: -
Recovery label received: -, Recovery label sent: 3
Resv style: 1 FF, Label in: 299920, Label out: 3
Time left: 141, Since: Mon Jun 10 13:10:11 2013
Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
Port number: sender 2 receiver 59956 protocol 0
PATH rcvfrom: 10.4.0.2 (ge-1/2/0.0) 105 pkts
Adspec: received MTU 1500 sent MTU 1500
PATH sentto: 10.2.0.1 (ge-1/2/1.0) 105 pkts
RESV rcvfrom: 10.2.0.1 (ge-1/2/1.0) 105 pkts
Explct route: 10.2.0.1
Record route: 10.4.0.2 <self> 10.2.0.1
Total 3 displayed, Up 3, Down 0

```

### Verifying BGP NRLI

**Purpose** Check the details of the BGP VPN network layer reachability information.

**Action** user@PE3> show bgp neighbor

```

Peer: 172.16.183.55+179 AS 64510 Local: 172.16.183.59+61747 AS 64510
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm    Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.183.59 Holdtime: 90 Preference: 170
NLRI configured with egress-protection: inet-vpn-unicast
Egress-protection NLRI inet-vpn-unicast, keep-import: [ remote-vrf ]
  Number of flaps: 0
  Peer ID: 172.16.183.55    Local ID: 172.16.183.59    Active Holdtime: 90
  Keepalive Interval: 30    Group index: 0    Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  NLRI of received end-of-rib markers: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 64510)
  Peer does not support Addpath
  Table bgp.13vpn.0
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not advertising
    Active prefixes:          0
    Received prefixes:        0
    Accepted prefixes:        0
    Suppressed due to damping: 0
  Last traffic (seconds): Received 25    Sent 21    Checked 11
  Input messages: Total 32046 Updates 7    Refreshes 0    Octets 609365
  Output messages: Total 32050 Updates 0    Refreshes 5    Octets 609010
  Output Queue[0]: 0

Peer: 172.16.183.56+62754 AS 64510 Local: 172.16.183.59+179 AS 64510
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm    Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.183.59 Holdtime: 90 Preference: 170
NLRI configured with egress-protection: inet-vpn-unicast
Egress-protection NLRI inet-vpn-unicast, keep-import: [ remote-vrf ]
  Number of flaps: 1
  Last flap event: TransportError
  Peer ID: 172.16.183.56    Local ID: 172.16.183.59    Active Holdtime: 90
  Keepalive Interval: 30    Group index: 0    Peer index: 1
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 64510)

```

```
Peer does not support Addpath
Table bgp.13vpn.0
  RIB State: BGP restart is complete
  RIB State: VPN restart is complete
  Send state: not advertising
  Active prefixes:          0
  Received prefixes:        0
  Accepted prefixes:        0
  Suppressed due to damping: 0
  Last traffic (seconds): Received 19   Sent 8   Checked 34
  Input messages:  Total 10025  Updates 0   Refreshes 2   Octets 190523
  Output messages: Total 10024  Updates 0   Refreshes 2   Octets 190504
  Output Queue[0]: 0
```

**Meaning** NLRI configured with `egress-protection` shows the BGP family configured with egress protection. `egress-protection NLRI inet-vpn-unicast, keep-import: [remote-vrf]` shows the egress protection routing policy for the BGP group.

---

### Verifying the Traffic Engineering Database

**Purpose** On all devices, check the TED.

**Action** user@PE1> show ted database

```

TED database: 9 ISIS nodes 5 INET nodes
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)                Rtr   44      3      3 IS-IS(2)
  To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
    Local interface index: 149, Remote interface index: 0
  To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
    Local interface index: 150, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
    Local interface index: 133, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.02                            Net   111      2      2 IS-IS(2)
  To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2-6.6.6.6.00(6.6.6.6)         Rtr   345      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 1, Remote interface index: 2147618817
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2, Remote interface index: 2147618818
ID                               Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)          Rtr   487      1      1 IS-IS(2)
  To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
    Local interface index: 148, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)          Rtr   353      3      3 IS-IS(2)
  To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
    Local interface index: 155, Remote interface index: 0
  To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
    Local interface index: 153, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618817, Remote interface index: 1
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.02                          Net    59      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)          Rtr   435      3      3 IS-IS(2)
  To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
    Local interface index: 154, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
    Local interface index: 158, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618818, Remote interface index: 2
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.02                          Net   706      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.03                          Net   583      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0

```

Local interface index: 0, Remote interface index: 0

user@PE2> show ted database

TED database: 9 ISIS nodes 5 INET nodes

```

ID                               Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)                Rtr   44      3      3 IS-IS(2)
  To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
  Local interface index: 150, Remote interface index: 0
  To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
  Local interface index: 149, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
  Local interface index: 133, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.02                             Net   111      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
  To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2-6.6.6.6.00(6.6.6.6)         Rtr   345      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 1, Remote interface index: 2147618817
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 2, Remote interface index: 2147618818
ID                               Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)           Rtr   487      1      1 IS-IS(2)
  To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
  Local interface index: 148, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)           Rtr   353      3      3 IS-IS(2)
  To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
  Local interface index: 155, Remote interface index: 0
  To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
  Local interface index: 153, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 2147618817, Remote interface index: 1
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.02                             Net    60      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)           Rtr   435      3      3 IS-IS(2)
  To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
  Local interface index: 154, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
  Local interface index: 158, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 2147618818, Remote interface index: 2
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.02                             Net   706      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.03                             Net   583      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
  Local interface index: 0, Remote interface index: 0

```

```

user@PE3> show ted database
TED database: 9 ISIS nodes 5 INET nodes
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)                Rtr    44    3    3 IS-IS(2)
  To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
    Local interface index: 149, Remote interface index: 0
  To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
    Local interface index: 150, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
    Local interface index: 133, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.02                            Net    111    2    2 IS-IS(2)
  To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2-6.6.6.6.00(6.6.6.6)         Rtr    345    2    2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 1, Remote interface index: 2147618817
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2, Remote interface index: 2147618818
ID                               Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)           Rtr    487    1    1 IS-IS(2)
  To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
    Local interface index: 148, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)           Rtr    353    3    3 IS-IS(2)
  To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
    Local interface index: 153, Remote interface index: 0
  To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
    Local interface index: 155, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618817, Remote interface index: 1
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.02                            Net    59    2    2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)           Rtr    435    3    3 IS-IS(2)
  To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
    Local interface index: 154, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
    Local interface index: 158, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618818, Remote interface index: 2
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.02                            Net    706    2    2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.03                            Net    583    2    2 IS-IS(2)
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
-----

```

```

user@P1> show ted database
TED database: 9 ISIS nodes 5 INET nodes
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)                 Rtr   44      3      3 IS-IS(2)
  To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
    Local interface index: 150, Remote interface index: 0
  To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
    Local interface index: 149, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
    Local interface index: 133, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
P1.02                             Net   111      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2-6.6.6.6.00(6.6.6.6)          Rtr   345      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 1, Remote interface index: 2147618817
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2, Remote interface index: 2147618818
ID                               Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)            Rtr   487      1      1 IS-IS(2)
  To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
    Local interface index: 148, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)            Rtr   353      3      3 IS-IS(2)
  To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
    Local interface index: 155, Remote interface index: 0
  To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
    Local interface index: 153, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618817, Remote interface index: 1
ID                               Type Age(s) LnkIn LnkOut Protocol
PE2.02                             Net    59      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)            Rtr   435      3      3 IS-IS(2)
  To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
    Local interface index: 154, Remote interface index: 0
  To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
    Local interface index: 158, Remote interface index: 0
  To: PE2-6.6.6.6.00(6.6.6.6), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 2147618818, Remote interface index: 2
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.02                             Net   706      2      2 IS-IS(2)
  To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
ID                               Type Age(s) LnkIn LnkOut Protocol
PE3.03                             Net   583      2      2 IS-IS(2)
  To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0
  To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
    Local interface index: 0, Remote interface index: 0

```

### Verifying the IS-IS Database

---

**Purpose** On all devices, check the IS-IS database.

**Action** user@PE1> show isis database

IS-IS level 1 link-state database:  
0 LSPs

IS-IS level 2 link-state database:

| LSP ID            | Sequence | Checksum | Lifetime | Attributes     |
|-------------------|----------|----------|----------|----------------|
| P1.00-00          | 0x46b    | 0x1924   | 590      | L1 L2          |
| P1.02-00          | 0x465    | 0xe67a   | 523      | L1 L2          |
| PE2-6.6.6.6.00-00 | 0xd0e    | 0x6b8d   | 1086     | L1 L2 Overload |
| PE1.00-00         | 0x46f    | 0xa8b    | 992      | L1 L2          |
| PE2.00-00         | 0x46b    | 0xefd6   | 1077     | L1 L2          |
| PE2.02-00         | 0x464    | 0x4db4   | 573      | L1 L2          |
| PE3.00-00         | 0x46f    | 0xb6e8   | 1016     | L1 L2          |
| PE3.02-00         | 0x465    | 0x2675   | 762      | L1 L2          |
| PE3.03-00         | 0x465    | 0x47b2   | 797      | L1 L2          |

9 LSPs

user@PE2> show isis database

IS-IS level 1 link-state database:  
0 LSPs

IS-IS level 2 link-state database:

| LSP ID            | Sequence | Checksum | Lifetime | Attributes     |
|-------------------|----------|----------|----------|----------------|
| P1.00-00          | 0x46b    | 0x1924   | 590      | L1 L2          |
| P1.02-00          | 0x465    | 0xe67a   | 523      | L1 L2          |
| PE2-6.6.6.6.00-00 | 0xd0e    | 0x6b8d   | 1090     | L1 L2 Overload |
| PE1.00-00         | 0x46f    | 0xa8b    | 988      | L1 L2          |
| PE2.00-00         | 0x46b    | 0xefd6   | 1080     | L1 L2          |
| PE2.02-00         | 0x464    | 0x4db4   | 576      | L1 L2          |
| PE3.00-00         | 0x46f    | 0xb6e8   | 1018     | L1 L2          |
| PE3.02-00         | 0x465    | 0x2675   | 763      | L1 L2          |
| PE3.03-00         | 0x465    | 0x47b2   | 799      | L1 L2          |

9 LSPs

user@PE3> show isis database

IS-IS level 1 link-state database:  
0 LSPs

IS-IS level 2 link-state database:

| LSP ID            | Sequence | Checksum | Lifetime | Attributes     |
|-------------------|----------|----------|----------|----------------|
| P1.00-00          | 0x46b    | 0x1924   | 590      | L1 L2          |
| P1.02-00          | 0x465    | 0xe67a   | 523      | L1 L2          |
| PE2-6.6.6.6.00-00 | 0xd0e    | 0x6b8d   | 1088     | L1 L2 Overload |
| PE1.00-00         | 0x46f    | 0xa8b    | 988      | L1 L2          |
| PE2.00-00         | 0x46b    | 0xefd6   | 1079     | L1 L2          |
| PE2.02-00         | 0x464    | 0x4db4   | 575      | L1 L2          |
| PE3.00-00         | 0x46f    | 0xb6e8   | 1020     | L1 L2          |
| PE3.02-00         | 0x465    | 0x2675   | 765      | L1 L2          |
| PE3.03-00         | 0x465    | 0x47b2   | 801      | L1 L2          |

9 LSPs

user@P1> show isis database

IS-IS level 1 link-state database:  
0 LSPs

IS-IS level 2 link-state database:

| LSP ID            | Sequence | Checksum | Lifetime | Attributes     |
|-------------------|----------|----------|----------|----------------|
| P1.00-00          | 0x46b    | 0x1924   | 592      | L1 L2          |
| P1.02-00          | 0x465    | 0xe67a   | 525      | L1 L2          |
| PE2-6.6.6.6.00-00 | 0xd0e    | 0x6b8d   | 1088     | L1 L2 Overload |
| PE1.00-00         | 0x46f    | 0xa8b    | 990      | L1 L2          |

|           |       |        |      |    |    |
|-----------|-------|--------|------|----|----|
| PE2.00-00 | 0x46b | 0xefd6 | 1079 | L1 | L2 |
| PE2.02-00 | 0x464 | 0x4db4 | 575  | L1 | L2 |
| PE3.00-00 | 0x46f | 0xb6e8 | 1018 | L1 | L2 |
| PE3.02-00 | 0x465 | 0x2675 | 763  | L1 | L2 |
| PE3.03-00 | 0x465 | 0x47b2 | 799  | L1 | L2 |
| 9 LSPs    |       |        |      |    |    |

- Related Documentation**
- <http://www.ethernetacademy.net/Ethernet-Academy-Articles/end-to-end-service-restoration>
  - [Egress Protection for Layer 3 VPN Edge Protection Overview on page 269](#)

## Example: Configuring Provider Edge Link Protection in Layer 3 VPNs

- [Understanding Provider Edge Link Protection in Layer 3 VPNs on page 316](#)
- [Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 317](#)

### Understanding Provider Edge Link Protection in Layer 3 VPNs

In an MPLS service provider network, a customer can have dual-homed CE routers that are connected to the service provider through different PE routers. This setup enables load balancing of traffic in the service provider network. However, this can lead to disruption in traffic if the link between a CE router and a PE router goes down. Hence, a precomputed protection path should be configured such that if a link between a CE router and a PE router goes down, the protection path (also known as the backup path) between the CE router and an alternate PE router can be used.

To configure a path to be a protection path, use the **protection** statement at the **[edit routing-instances *instance-name* protocols bgp family inet unicast]** hierarchy level:

```
routing-instances {
  customer {
    instance-type vrf;
    ...
    protocols {
      bgp {
        type external;
        ...
        family inet {
          unicast {
            protection;
          }
        }
        family inet6 {
          unicast {
            protection;
          }
        }
      }
    }
  }
}
```

The **protection** statement indicates that protection is required on prefixes received from the particular neighbor or family. After protection is enabled for a given family, group, or

neighbor, protection entries are added for prefixes or next hops received from the given peer.



**NOTE:** A protection path can be selected only if the best path has already been installed by BGP in the forwarding table. This is because a protection path cannot be used as the best path.



**NOTE:** The option `vrf-table-label` must be configured under the `[routing-instances instance-name]` hierarchy for the routers that have protected PE-CE links. This applies to Junos OS Releases 12.3 through 13.2 inclusive.

The protection path selection takes place based on the value of two state flags:

- The **ProtectionPath** flag indicates paths requesting protection.
- The **ProtectionCand** flag indicates the route entry that can be used as a protection path.



**NOTE:**

- Provider edge link protection is configured only for external peers.
- If provider edge link protection is configured with the `equal-external-internal` multipath statement, multipath takes precedence over protection.

## Example: Configuring Provider Edge Link Protection in Layer 3 VPNs

This example shows how to configure a provider edge protection path that can be used in case of a link failure in an MPLS network.

- [Requirements on page 317](#)
- [Overview on page 318](#)
- [Configuration on page 318](#)
- [Verification on page 328](#)

### Requirements

This example uses the following hardware components, software components and configuration options:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 12.3 through 13.2 inclusive
- The option `vrf-table-label` must be enabled at the `[routing-instances instance-name]` hierarchy level for routers with protected PE-CE links.

## Overview

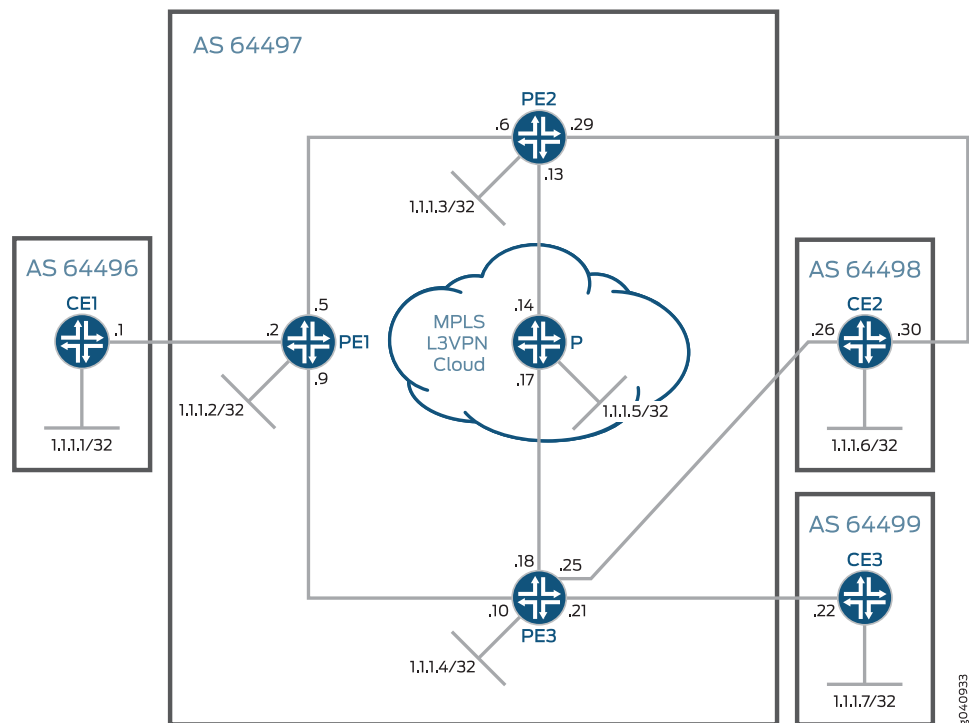
The following example shows how to configure provider edge link protection in a Layer 3 VPN.

### Topology

In this example, a Layer 3 VPN is set up by configuring three customer edge devices and three service provider edge devices in four autonomous systems. The CE devices are configured in AS 64496, AS 64498, and AS 64499. The PE devices are configured in AS 64497.

Figure 14 on page 62 shows the topology used in this example.

Figure 39: Provider Edge Link Protection in a Layer 3 VPN



The aim of this example is to protect the provider edge link between Routers PE2 and CE2. Protection is configured on the backup link between Routers PE3 and CE2, such that the traffic can be routed through this link when the PE2-CE2 link goes down.

## Configuration

### CLI Quick Configuration

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

```
Router CE1
set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
```

```

set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.1/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::1/128
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 64496
set protocols bgp group toPE1 type external
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 peer-as 64497
set protocols bgp group toPE1 neighbor 10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

```

Router PE1 set interfaces ge-2/0/0 unit 0 description toCE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.2/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE2
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.5/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toPE3
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.9/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.2/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::2/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.2
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/0.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE1 type external
set routing-instances radium protocols bgp group toCE1 peer-as 64496
set routing-instances radium protocols bgp group toCE1 neighbor 10.1.1.1
set policy-options policy-statement lb then load-balance per-packet

```

```

Router PE2 set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.6/30

```

```

set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.13/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.29/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.3/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::3/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.3
set protocols bgp group toInternal neighbor 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.4
set routing-options router-id 1.1.1.3
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.30
set policy-options policy-statement lb then load-balance per-packet

```

#### Router PE3

```

set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.10/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.18/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.25/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces ge-2/0/3 unit 0 description toCE3
set interfaces ge-2/0/3 unit 0 family inet address 10.1.1.21/30
set interfaces ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
set interfaces ge-2/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.4/32

```

```

set interfaces lo0 unit 0 family inet6 address 2001:db8::4/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 1.1.1.4
set protocols bgp group toInternal neighbor 1.1.1.2
set protocols bgp group toInternal neighbor 1.1.1.3
set routing-options router-id 1.1.1.4
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium vrf-table-label
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium interface ge-2/0/3.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.26
set routing-instances radium protocols bgp group toCE2 family inet unicast protection
set routing-instances radium protocols bgp group toCE2 family inet6 unicast protection
set routing-instances radium protocols bgp group toCE3 type external
set routing-instances radium protocols bgp group toCE3 peer-as 64499
set routing-instances radium protocols bgp group toCE3 neighbor 10.1.1.22
set policy-options policy-statement lb then load-balance per-packet

```

**Router P**

```

set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.14/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.17/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.5/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::5/128
set routing-options router-id 1.1.1.5
set routing-options autonomous-system 64497
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5

```

**Router CE2**

```

set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.30/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.26/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.6/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::6/128
set routing-options router-id 1.1.1.6
set routing-options autonomous-system 64498
set protocols bgp group toAS2 type external
set protocols bgp group toAS2 export send-direct
set protocols bgp group toAS2 peer-as 64497
set protocols bgp group toAS2 neighbor 10.1.1.25
set protocols bgp group toAS2 neighbor 10.1.1.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

**Router CE3**

```

set interfaces ge-2/0/0 unit 0 description toPE3
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.22/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.7/32
set interfaces lo0 unit 0 family inet6 address 2001:db8::7/128
set routing-options router-id 1.1.1.7
set routing-options autonomous-system 64499
set protocols bgp group toPE3 type external
set protocols bgp group toPE3 export send-direct
set protocols bgp group toPE3 peer-as 64497
set protocols bgp group toPE3 neighbor 10.1.1.21
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept

```

### *Configuring Provider Edge Link Protection in Layer 3 VPNs*

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

To configure provider edge link protection:

1. Configure the router interfaces.

```

[edit interfaces]
user@PE3# set interfaces ge-2/0/0 unit 0 description toPE1
user@PE3# set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.10/30
user@PE3# set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64
eui-64
user@PE3# set interfaces ge-2/0/0 unit 0 family mpls

user@PE3# set interfaces ge-2/0/1 unit 0 description toP
user@PE3# set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.18/30
user@PE3# set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64
eui-64

```

```

user@PE3# set interfaces ge-2/0/1 unit 0 family mpls
user@PE3# set interfaces ge-2/0/2 unit 0 description toCE2
user@PE3# set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.25/30
user@PE3# set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:25::/64
eui-64
user@PE3# set interfaces ge-2/0/2 unit 0 family mpls
user@PE3# set interfaces ge-2/0/3 unit 0 description toCE3
user@PE3# set interfaces ge-2/0/3 unit 0 family inet address 10.1.1.21/30
user@PE3# set interfaces ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64
eui-64
user@PE3# set interfaces ge-2/0/3 unit 0 family mpls
user@PE3# set interfaces lo0 unit 0 family inet address 1.1.1.4/32
user@PE3# set interfaces lo0 unit 0 family inet6 address 2001:db8::4/128

```

Similarly, configure the interfaces on all other routers.

2. Configure the router ID and autonomous system (AS) number.

```

[edit routing-options]
user@PE3# set router-id 1.1.1.4
user@PE3# set autonomous-system 64497

```

Similarly, configure the router ID and AS number for all other routers. In this example, the router ID is chosen to be identical to the loopback address configured on the router.

3. Configure MPLS and LDP on all interfaces of Router PE3.

```

[edit protocols]
user@PE3# set mpls interface all
user@PE3# set ldp interface all

```

Similarly, configure other PE routers.

4. Configure an IGP on the core-facing interfaces of Router PE3.

```

[edit protocols ospf area 0.0.0.0]
user@PE3# set interface lo0.0 passive
user@PE3# set interface ge-2/0/1.0 metric 5
user@PE3# set interface ge-2/0/0.0 metric 10

[edit protocols ospf3 area 0.0.0.0]
user@PE3# set interface lo0.0 passive
user@PE3# set interface ge-2/0/1.0 metric 5
user@PE3# set interface ge-2/0/0.0 metric 10

```

Similarly, configure other PE routers.

5. Configure a policy that exports the routes from the routing table into the forwarding table on Router PE3.

```

[edit policy-options]
user@PE3# set policy-statement lb then load-balance per-packet

[edit routing-options]
user@PE3# set forwarding-table export lb

```

Similarly, configure other PE routers.

6. Configure BGP on Router CE2, and include a policy for exporting routes to and from the service provider network.

```
[edit policy-options]
user@CE2# set policy-statement send-direct from protocol direct
user@CE2# set policy-statement send-direct then accept

[edit protocols bgp group toAS2]
user@CE2# set type external
user@CE2# set export send-direct
user@CE2# set peer-as 64497
user@CE2# set neighbor 10.1.1.25
user@CE2# set neighbor 10.1.1.29
```

Similarly, configure other CE routers.

7. Configure BGP on Router PE3 for routing within the provider core.

```
[edit protocols bgp group toInternal]
user@PE3# set type internal
user@PE3# set family inet-vpn unicast
user@PE3# set family inet6-vpn unicast
user@PE3# set multipath
user@PE3# set local-address 1.1.1.4
user@PE3# set neighbor 1.1.1.2
user@PE3# set neighbor 1.1.1.3
```

Similarly, configure other PE routers.

8. Configure the Layer 3 VPN routing instance on Router PE3.

```
[set routing-instances radium]
user@PE3# set instance-type vrf
user@PE3# set vrf-table-label
user@PE3# set interface ge-2/0/2.0
user@PE3# set interface ge-2/0/3.0
user@PE3# set route-distinguisher 64497:1
user@PE3# set vrf-target target:64497:1

[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set type external
user@PE3# set peer-as 64498
user@PE3# set neighbor 10.1.1.26

[edit routing-instances radium protocols bgp group toCE3]
user@PE3# set type external
user@PE3# set peer-as 64499
user@PE3# set neighbor 10.1.1.22
```

Similarly, configure other PE routers.

9. Configure provider edge link protection on the link between Routers PE3 and CE2.

```
[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set family inet unicast protection
user@PE3# set family inet6 unicast protection
```

### Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show policy-options**, **show protocols**, and **show routing-instances** commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show interfaces
ge-2/0/0 {
  unit 0 {
    description toPE1;
    family inet {
      address 10.1.1.10/30;
    }
    family inet6 {
      address 2001:db8:0:9::/64 {
        eui-64;
      }
    }
    family mpls;
  }
}

ge-2/0/1 {
  unit 0 {
    description toP;
    family inet {
      address 10.1.1.18/30;
    }
    family inet6 {
      address 2001:db8:0:17::/64 {
        eui-64;
      }
    }
    family mpls;
  }
}

ge-2/0/2 {
  unit 0 {
    description toCE2;
    family inet {
      address 10.1.1.25/30;
    }
    family inet6 {
      address 2001:db8:0:25::/64 {
        eui-64;
      }
    }
    family mpls;
  }
}

ge-2/0/3 {
  unit 0 {
    description toCE3;
    family inet {
      address 10.1.1.21/30;
    }
  }
}
```

```
    }
    family inet6 {
        address 2001:db8:0:21::/64 {
            eui-64;
        }
    }
    family mpls;
}
}
```

```
lo0 {
    unit 0 {
        family inet {
            address 1.1.1.4/32;
        }
        family inet6 {
            address 2001:db8::4/128;
        }
    }
}
```

**user@PE3# show routing-options**

```
router-id 1.1.1.4;
autonomous-system 64497;
forwarding-table {
    export lb;
}
```

**user@PE3# show policy-options**

```
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
```

**user@PE3# show protocols**

```
mps {
    interface all;
}
bgp {
    group toInternal {
        type internal;
        local-address 1.1.1.4;
        family inet-vpn {
            unicast;
        }
        family inet6-vpn {
            unicast;
        }
        multipath;
        neighbor 1.1.1.2;
        neighbor 1.1.1.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
    }
}
```

```

    }
    interface ge-2/0/0.0 {
        metric 10;
    }
}
ospf3 {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
        interface ge-2/0/0.0 {
            metric 10;
        }
    }
}
ldp {
    interface all;
}

user@PE3# show routing-instances
radium {
    instance-type vrf;
    interface ge-2/0/2.0;
    interface ge-2/0/3.0;
    route-distinguisher 64497:1;
    vrf-target target:64497:1;
    protocols {
        bgp {
            group toCE2 {
                type external;
                family inet {
                    unicast {
                        protection;
                    }
                }
                family inet6 {
                    unicast {
                        protection;
                    }
                }
                peer-as 64498;
                neighbor 10.1.1.26;
            }
            group toCE3 {
                type external;
                peer-as 64499;
                neighbor 10.1.1.22;
            }
        }
    }
}

```

Run these commands on all other routers to confirm the configurations. If you are done configuring the routers, enter **commit** from configuration mode.

## Verification

Confirm that the configuration is working properly.

- [Verifying BGP on page 328](#)
- [Verifying Provider Edge Link Protection on page 329](#)

### Verifying BGP

**Purpose** Verify that BGP is functional in the Layer 3 VPN.

**Action** From operational mode on Router PE3, run the **show route protocol bgp** command.

```
user@PE3> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32      *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                AS path: 64496 I, validation-state: unverified
                > to 10.1.1.9 via ge-2/0/0.0, Push 299792
1.1.1.6/32      @[BGP/170] 00:09:40, localpref 100
                AS path: 64498 I, validation-state: unverified
                > to 10.1.1.26 via ge-2/0/2.0
                [BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                AS path: 64498 I, validation-state: unverified
                > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
1.1.1.7/32      *[BGP/170] 00:09:26, localpref 100
                AS path: 64499 I, validation-state: unverified
                > to 10.1.1.22 via ge-2/0/3.0
10.1.1.0/30     *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                AS path: I, validation-state: unverified
                > to 10.1.1.9 via ge-2/0/0.0, Push 299792
10.1.1.20/30    [BGP/170] 00:09:26, localpref 100
                AS path: 64499 I, validation-state: unverified
                > to 10.1.1.22 via ge-2/0/3.0
10.1.1.24/30    [BGP/170] 00:09:40, localpref 100
                AS path: 64498 I, validation-state: unverified
                > to 10.1.1.26 via ge-2/0/2.0
10.1.1.28/30    *[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
                AS path: I, validation-state: unverified
                > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)

                [BGP/170] 00:09:40, localpref 100
                AS path: 64498 I, validation-state: unverified
                > to 10.1.1.26 via ge-2/0/2.0

mpls.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

64497:1:1.1.1.1/32
                *[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
                AS path: 64496 I, validation-state: unverified
```

```

> to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:1.1.1.6/32
*[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
  AS path: 64498 I, validation-state: unverified
> to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
64497:1:10.1.1.0/30
*[BGP/170] 00:09:15, localpref 100, from 1.1.1.2
  AS path: I, validation-state: unverified
> to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:10.1.1.28/30
*[BGP/170] 00:09:07, localpref 100, from 1.1.1.3
  AS path: I, validation-state: unverified
> to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)

```

inet6.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

radium.inet6.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)

The output shows all the BGP routes in the routing table of Router PE3. This indicates that BGP is functioning as required.

Similarly, run this command on other routers to check if BGP is operational.

**Meaning** BGP is functional in the Layer 3 VPN.

### *Verifying Provider Edge Link Protection*

**Purpose** Verify that the provider edge link between Routers PE2 and CE2 is protected.

**Action** To verify that provider edge link protection is configured correctly:

1. Confirm that a route on Router CE2 is advertised to Router PE3, directly and through Router PE2.

If the route is advertised correctly, you will see multiple paths for the route.

From operational mode on Router PE3, run the **show route destination-prefix** command.

```

user@PE3> show route 1.1.1.6
radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

1.1.1.6/32
@[BGP/170] 02:55:36, localpref 100
  AS path: 64498 I, validation-state: unverified
> to 10.1.1.26 via ge-2/0/2.0
  [BGP/170] 00:10:13, localpref 100, from 1.1.1.3
    AS path: 64498 I, validation-state: unverified
> to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)

#[Multipath/255] 00:10:13
> to 10.1.1.26 via ge-2/0/2.0
  to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)

```

The output verifies the presence of multiple paths from Router PE3 to the destination route, **1.1.1.6**, on Router CE2. The first path is directly through the PE3-CE2 link (**10.1.1.26**). The second path is through the provider core and PE2 (**10.1.1.17**).

2. Verify that the protection path is correctly configured by confirming that the weight for the active path being protected is **0x1**, and the weight for the protection candidate path is **0x4000**.

From operational mode on Router PE3, run the **show route destination-prefix extensive** command.

```

user@PE3> show route 1.1.1.6 extensive
radius.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
1.1.1.6/32 (3 entries, 2 announced)
    State: <CalcForwarding>
TSI:
KRT in-kernel 1.1.1.6/32 -> {list:10.1.1.26, indirect(1048584)}
Page 0 idx 1 Type 1 val 9229c38
    Nexthop: Self
    AS path: [64497] 64498 I
    Communities:
Page 0 idx 2 Type 1 val 9229cc4
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [64497] 64498 I
    Communities: target:64497:1
Path 1.1.1.6 from 10.1.1.26 Vector len 4. Val: 1 2
    @BGP    Preference: 170/-101
            Next hop type: Router, Next hop index: 994
            Address: 0x9240a74
            Next-hop reference count: 5
            Source: 10.1.1.26
            Next hop: 10.1.1.26 via ge-2/0/2.0, selected
            Session Id: 0x200001
            State: <Active Ext ProtectionPath ProtectionCand>
            Peer AS: 64498
            Age: 2:55:54
            Validation State: unverified
            Task: BGP_64498.10.1.1.26+52214
            Announcement bits (1): 2-BGP_RT_Background
            AS path: 64498 I
            Accepted
            Localpref: 100
            Router ID: 1.1.1.6
    BGP    Preference: 170/-101
            Route Distinguisher: 64497:1
            Next hop type: Indirect
            Address: 0x92413a8
            Next-hop reference count: 6
            Source: 1.1.1.3
            Next hop type: Router, Next hop index: 1322
            Next hop: 10.1.1.17 via ge-2/0/1.0, selected
            Label operation: Push 299840, Push 299776(top)
            Label TTL action: prop-ttl, prop-ttl(top)
            Session Id: 0x200005
            Protocol next hop: 1.1.1.3
            Push 299840
            Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
            State: <Secondary NotBest Int Ext ProtectionCand>
            Inactive reason: Not Best in its group - Interior > Exterior > Exterior via
Interior
            Local AS: 64497 Peer AS: 64497
            Age: 10:31      Metric2: 1

```

```

Validation State: unverified
Task: BGP_64497.1.1.1.3+179
Local AS: 64497 Peer AS: 64497
Age: 10:31 Metric2: 1
Validation State: unverified
Task: BGP_64497.1.1.1.3+179
AS path: 64498 I
Communities: target:64497:1
Import Accepted
VPN Label: 299840
Localpref: 100
Router ID: 1.1.1.3
Primary Routing Table bgp.13vpn.0
Indirect next hops: 1
  Protocol next hop: 1.1.1.3 Metric: 1
  Push 299840
  Indirect next hop: 94100ec 1048584 INH Session ID:
0x20000b
    Indirect path forwarding next hops: 1
      Next hop type: Router
      Next hop: 10.1.1.17 via ge-2/0/1.0
      Session Id: 0x200005
    1.1.1.3/32 Originating RIB: inet.3
      Metric: 1 Node path count: 1
      Forwarding nexthops: 1
        Nexthop: 10.1.1.17 via ge-2/0/1.0
#Multipath Preference: 255
Next hop type: List, Next hop index: 1048585
Address: 0x944c154
Next-hop reference count: 2
Next hop: ELNH Address 0x9240a74 weight 0x1, selected
equal-external-internal-type external
  Next hop type: Router, Next hop index: 994
  Address: 0x9240a74
  Next-hop reference count: 5
  Next hop: 10.1.1.26 via ge-2/0/2.0
Next hop: ELNH Address 0x92413a8 weight 0x4000
equal-external-internal-type internal
  Next hop type: Indirect
  Address: 0x92413a8
  Next-hop reference count: 6
  Protocol next hop: 1.1.1.3
  Push 299840
  Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
    Next hop type: Router, Next hop index: 1322
    Address: 0x9241310
    Next-hop reference count: 4
    Next hop: 10.1.1.17 via ge-2/0/1.0
    Label operation: Push 299840, Push 299776(top)
    Label TTL action: prop-ttl, prop-ttl(top)
State: <ForwardingOnly Int Ext>
Inactive reason: Forwarding use only
Age: 10:31
Validation State: unverified
Task: RT
Announcement bits (1): 0-KRT
AS path: 64498 I

```

The output shows that the weight (**0x4000**) assigned to the PE3-CE2 path is greater than the weight (**0x1**) assigned to the PE2-CE2 path. This confirms that the PE2-CE2 path is protected by the PE3-CE2 path.

**Meaning** The provider edge link between Routers PE2 and CE2 is protected.

## Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths

---

This example shows how to configure a labeled unicast protection path that can be used in case of a link failure in a carrier-of-carriers topology.

- [Requirements on page 332](#)
- [Overview on page 332](#)
- [Configuration on page 333](#)
- [Verification on page 344](#)

### Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 13.3 or later

### Overview

This example shows how to configure labeled-unicast link protection in a Layer 3 VPN.

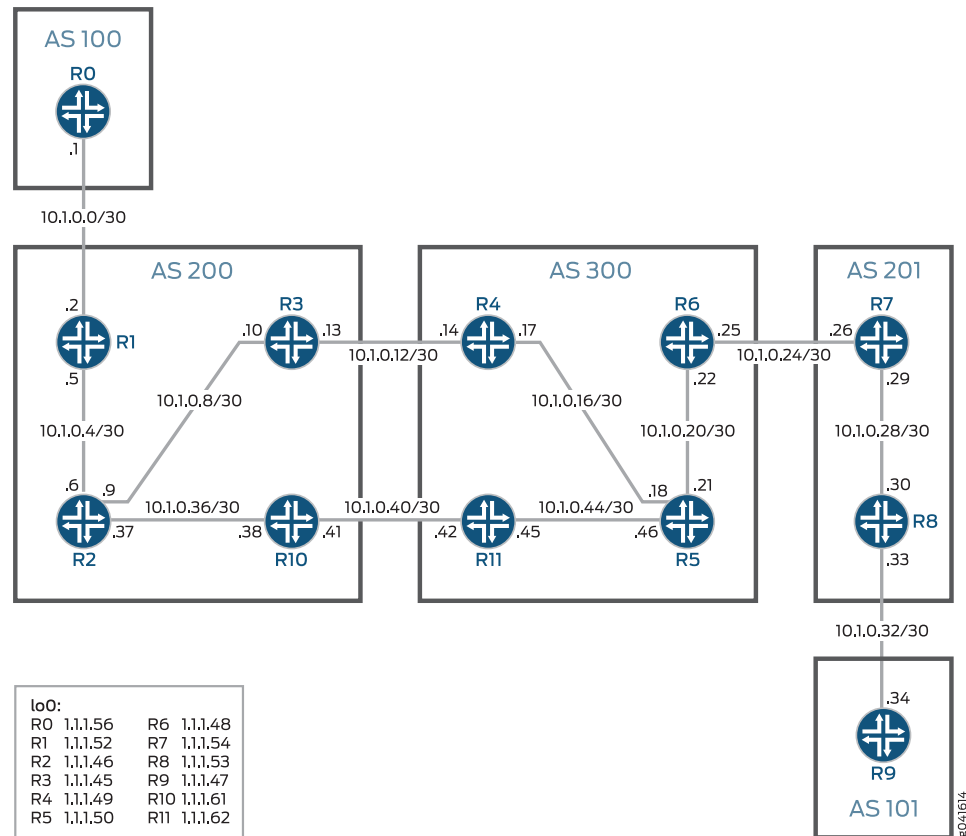
#### Topology

---

In this example, a carrier-of-carriers topology is set up by configuring two customer edge devices and eight service provider edge devices in five autonomous systems. The CE devices are configured in AS100 and AS101. The PE devices are configured in AS200, AS300, and AS201.

[Figure 14 on page 62](#) shows the topology used in this example.

Figure 40: Labeled Unicast Link Protection in a Layer 3 VPN



The aim of this example is to protect the provider edge link between Routers R4 and R3. Protection is configured on the primary link between R4 and R3 such that the traffic can be routed through the backup link (R11 to R10) when the primary link goes down.



**NOTE:** Protection can also be configured on the secondary link between R11 and R10 so that if that link becomes the primary link and the R4-R3 link becomes secondary, the R11-R10 link will be protected as well.

## Configuration

### CLI Quick Configuration

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



**NOTE:** Protection is added to the configuration only after the initial configuration is committed and BGP has installed the best path in the forwarding table.

**Router R0**

```
set interfaces ge-2/0/0 unit 0 description toR1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.1/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.56/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2056.00
set routing-options router-id 1.1.1.56
set routing-options autonomous-system 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
```

**Router R1**

```
set interfaces ge-2/0/0 unit 0 description toR0
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.2/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR2
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.5/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.52/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2052.00
set routing-options router-id 1.1.1.52
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols bgp group toR8 local-address 1.1.1.52
set protocols bgp group toR8 type external
set protocols bgp group toR8 multihop ttl 10
set protocols bgp group toR8 family inet-vpn unicast
set protocols bgp group toR8 neighbor 1.1.1.53 peer-as 201
set policy-options policy-statement child_vpn_routes from protocol bgp
set policy-options policy-statement child_vpn_routes then accept
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-2/0/0.0
set routing-instances customer-provider-vpn route-distinguisher 1.1.1.45:1
set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
  ge-2/0/0.0
```

**Router R2**

```
set interfaces ge-2/0/0 unit 0 description toR1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.6/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.9/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
```

```

set interfaces ge-2/0/2 unit 0 description toR10
set interfaces ge-2/0/2 unit 0 family inet address 10.1.0.37/30
set interfaces ge-2/0/2 unit 0 family iso
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.46/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2046.00
set routing-options router-id 1.1.1.46
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface ge-2/0/2.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/2.0 metric 10

```

**Router R3**

```

set interfaces ge-2/0/0 unit 0 description toR2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.10/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR4
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.13/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.45/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2045.00
set routing-options router-id 1.1.1.45
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR4 type external
set protocols bgp group toR4 import send-local
set protocols bgp group toR4 family inet labeled-unicast
set protocols bgp group toR4 export send-local
set protocols bgp group toR4 neighbor 10.1.0.14 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-local term 2 from metric 100
set policy-options policy-statement send-local term 2 then reject
set policy-options policy-statement send-local then accept

```

```
Router R4    set interfaces ge-2/0/0 unit 0 description toR3
              set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.14/30
              set interfaces ge-2/0/0 unit 0 family iso
              set interfaces ge-2/0/0 unit 0 family mpls
              set interfaces ge-2/0/1 unit 0 description toR5
              set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.17/30
              set interfaces ge-2/0/1 unit 0 family iso
              set interfaces ge-2/0/1 unit 0 family mpls
              set interfaces lo0 unit 0 family inet address 1.1.1.49/32
              set interfaces lo0 unit 0 family iso address
                47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
              set policy-options policy-statement 1b then load-balance per-packet
              set routing-options router-id 1.1.1.49
              set routing-options autonomous-system 300
              set routing-options forwarding-table export 1b
              set protocols mpls interface all
              set protocols ldp track-igp-metric
              set protocols ldp interface ge-2/0/1.0
              set protocols ldp interface lo0.0
              set protocols isis level 1 disable
              set protocols isis level 2 wide-metrics-only
              set protocols isis interface ge-2/0/1.0 level 2 metric 10
              set protocols isis interface lo0.0 passive
              set protocols bgp group parent-vpn-peers type internal
              set protocols bgp group parent-vpn-peers local-address 1.1.1.49
              set protocols bgp group parent-vpn-peers family inet-vpn unicast
              set protocols bgp group parent-vpn-peers neighbor 1.1.1.48
              set protocols bgp group parent-vpn-peers neighbor 1.1.1.62
              set routing-instances coc-provider-vpn instance-type vrf
              set routing-instances coc-provider-vpn interface ge-2/0/0.0
              set routing-instances coc-provider-vpn interface ge-2/0/2.0
              set routing-instances coc-provider-vpn route-distinguisher 1.1.1.49:1
              set routing-instances coc-provider-vpn vrf-target target:300:1
              set routing-instances coc-provider-vpn protocols bgp group toR3 type external
              set routing-instances coc-provider-vpn protocols bgp group toR3 family inet
                labeled-unicast per-prefix-label
              set routing-instances coc-provider-vpn protocols bgp group toR3 neighbor 10.1.0.13
                peer-as 200

Router R5    set interfaces ge-2/0/0 unit 0 description toR4
              set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.18/30
              set interfaces ge-2/0/0 unit 0 family iso
              set interfaces ge-2/0/0 unit 0 family mpls
              set interfaces ge-2/0/1 unit 0 description toR6
              set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.21/30
              set interfaces ge-2/0/1 unit 0 family iso
              set interfaces ge-2/0/1 unit 0 family mpls
              set interfaces ge-2/0/2 unit 0 description toR11
              set interfaces ge-2/0/2 unit 0 family inet address 10.1.0.46/30
              set interfaces ge-2/0/2 unit 0 family iso
              set interfaces ge-2/0/2 unit 0 family mpls
              set interfaces lo0 unit 0 family inet address 1.1.1.50/32
              set interfaces lo0 unit 0 family iso address
                47.0005.80ff.f800.0000.0108.0001.0102.5507.2050.00
              set routing-options router-id 1.1.1.50
```

```

set routing-options autonomous-system 300
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface ge-2/0/2.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/0.0 level 2 metric 10
set protocols isis interface ge-2/0/1.0 level 2 metric 10
set protocols isis interface ge-2/0/2.0 level 2 metric 10
set protocols isis interface lo0.0 passive

```

```

Router R6  set interfaces ge-2/0/0 unit 0 description toR5
            set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.22/30
            set interfaces ge-2/0/0 unit 0 family iso
            set interfaces ge-2/0/0 unit 0 family mpls
            set interfaces ge-2/0/1 unit 0 description toR7
            set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.25/30
            set interfaces ge-2/0/1 unit 0 family iso
            set interfaces ge-2/0/1 unit 0 family mpls
            set interfaces lo0 unit 0 family inet address 1.1.1.48/32
            set interfaces lo0 unit 0 family iso address
              47.0005.80ff.f800.0000.0108.0001.0102.5507.2048.00
            set routing-options router-id 1.1.1.48
            set routing-options autonomous-system 300
            set protocols mpls interface all
            set protocols ldp track-igp-metric
            set protocols ldp interface ge-2/0/0.0
            set protocols ldp interface lo0.0
            set protocols isis level 1 disable
            set protocols isis level 2 wide-metrics-only
            set protocols isis interface ge-2/0/0.0 level 2 metric 10
            set protocols isis interface lo0.0 passive
            set protocols bgp group parent-vpn-peers type internal
            set protocols bgp group parent-vpn-peers local-address 1.1.1.48
            set protocols bgp group parent-vpn-peers family inet-vpn unicast
            set protocols bgp group parent-vpn-peers neighbor 1.1.1.49
            set protocols bgp group parent-vpn-peers neighbor 1.1.1.62
            set routing-instances coc-provider-vpn instance-type vrf
            set routing-instances coc-provider-vpn interface ge-2/0/1.0
            set routing-instances coc-provider-vpn route-distinguisher 1.1.1.48:1
            set routing-instances coc-provider-vpn vrf-target target:300:1
            set routing-instances coc-provider-vpn protocols bgp group toR7 family inet
              labeled-unicast per-prefix-label
            set routing-instances coc-provider-vpn protocols bgp group toR7 type external
            set routing-instances coc-provider-vpn protocols bgp group toR7 neighbor 10.1.0.26
              peer-as 201

Router R7  set interfaces ge-2/0/0 unit 0 description toR6
            set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.26/30
            set interfaces ge-2/0/0 unit 0 family iso
            set interfaces ge-2/0/0 unit 0 family mpls
            set interfaces ge-2/0/1 unit 0 description toR8

```

```
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.29/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.54/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2054.00
set routing-options router-id 1.1.1.54
set routing-options autonomous-system 201
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols bgp group toR6 type external
set protocols bgp group toR6 import send-all
set protocols bgp group toR6 family inet labeled-unicast
set protocols bgp group toR6 export send-all
set protocols bgp group toR6 neighbor 10.1.0.25 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-all then accept
```

**Router R8**

```
set interfaces ge-2/0/0 unit 0 description toR7
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.30/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR9
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.33/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.53/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2053.00
set routing-options router-id 1.1.1.53
set routing-options autonomous-system 201
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR1 local-address 1.1.1.53
set protocols bgp group toR1 type external
set protocols bgp group toR1 multihop ttl 10
set protocols bgp group toR1 family inet-vpn unicast
set protocols bgp group toR1 neighbor 1.1.1.52 peer-as 200
set policy-options policy-statement child_vpn_routes from protocol bgp
set policy-options policy-statement child_vpn_routes then accept
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-2/0/1.0
set routing-instances customer-provider-vpn route-distinguisher 1.1.1.53:1
```

```

set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
  ge-2/0/1.0

```

**Router R9**

```

set interfaces ge-2/0/0 unit 0 description toR8
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.34/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.47/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2047.00
set routing-options router-id 1.1.1.47
set routing-options autonomous-system 101
set routing-options static route 11.11.11.11/32 discard
set protocols ospf export statics
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set policy-options policy-statement statics from route-filter 11.11.11.11/32 exact
set policy-options policy-statement statics then accept

```

**Router R10**

```

set interfaces ge-2/0/0 unit 0 description toR2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.38/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR11
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.41/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.61/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2061.00
set routing-options router-id 1.1.1.61
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR4 type external
set protocols bgp group toR4 import send-local
set protocols bgp group toR4 export send-local
set protocols bgp group toR4 neighbor 10.1.0.42 peer-as 300
set protocols bgp group toR4 inactive: neighbor 10.1.0.50 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-local term 2 from metric 100
set policy-options policy-statement send-local term 2 then reject

```

```
set policy-options policy-statement send-local then accept
```

```
Router R11
set interfaces ge-2/0/0 unit 0 description toR10
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.42/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR5
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.45/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.62/32
set interfaces lo0 unit 0 family iso address
  47.0005.80ff.f800.0000.0108.0001.0102.5507.2062.00
set routing-options router-id 1.1.1.62
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.62
set protocols bgp group parent-vpn-peers family inet-vpn unicast
set protocols bgp group parent-vpn-peers neighbor 1.1.1.48
set protocols bgp group parent-vpn-peers neighbor 1.1.1.62
set routing-instances coc-provider-vpn instance-type vrf
set routing-instances coc-provider-vpn interface ge-2/0/0.0
set routing-instances coc-provider-vpn route-distinguisher 1.1.1.62:1
set routing-instances coc-provider-vpn vrf-target target:300:1
set routing-instances coc-provider-vpn protocols bgp group toR10 family inet
  labeled-unicast per-prefix-label
set routing-instances coc-provider-vpn protocols bgp group toR10 type external
set routing-instances coc-provider-vpn protocols bgp group toR10 neighbor 10.1.0.41
  peer-as 200
```

### Configuring Provider Edge Link Protection in Layer 3 VPNs

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

To configure labeled unicast link protection:

1. Configure the router interfaces.

```
[edit interfaces]
user@R4# set ge-2/0/0 unit 0 description toR3
user@R4# set ge-2/0/0 unit 0 family inet address 10.1.0.14/30
user@R4# set ge-2/0/0 unit 0 family iso
user@R4# set ge-2/0/0 unit 0 family mpls

user@R4# set ge-2/0/1 unit 0 description toR5
user@R4# set ge-2/0/1 unit 0 family inet address 10.1.0.17/30
```

```

user@R4# set ge-2/0/1 unit 0 family iso
user@R4# set ge-2/0/1 unit 0 family mpls

user@R4# set lo0 unit 0 family inet address 1.1.1.49/32
user@R4# set lo0 unit 0 family iso address
47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00

```

Similarly, configure the interfaces on all other routers.

2. Configure the routing policy options on R4.

```

[edit policy-options]
user@R4# set policy-statement 1b then load-balance per-packet

```

Similarly, configure the policy options on routers R1, R3, R7, R8, R9, and R10 for this example.

3. Configure the router ID, autonomous system (AS) number, and any other routing options.

```

[edit routing-options]
user@R4# set router-id 1.1.1.49
user@R4# set autonomous-system 300
user@R4# set forwarding-table export 1b

```

Similarly, configure the router ID, AS number, and any other routing options for all other routers. In this example, the router ID is chosen to be identical to the loopback address configured on the router.

4. Configure MPLS and LDP on Router R4.

```

[edit protocols]
user@R4# set mpls interface all
user@R4# set ldp track-igp-metric
user@R4# set ldp interface ge-2/0/1.0
user@R4# set ldp interface lo0.0

```

Similarly, configure MPLS and LDP on all other routers except R0 and R9.

5. Configure an IGP on the core-facing interfaces of Router R4.

```

[edit protocols isis]
user@R4# set level 1 disable
user@R4# set level 2 wide-metrics-only
user@R4# set interface ge-2/0/1.0 level 2 metric 10
user@R4# set interface lo0.0 passive

```

Similarly, configure other routers (IS-IS on R5, R6, and R11 and OSPF on all other routers in this example).

6. Configure BGP on Router R4.

```

[edit protocols bgp group parent-vpn-peers]
user@R4# set type internal
user@R4# set local-address 1.1.1.49
user@R4# set family inet-vpn unicast
user@R4# set neighbor 1.1.1.48
user@R4# set neighbor 1.1.1.62

```

Similarly, configure BGP on routers R1, R3, R6, R7, R8, R10, and R11.

- Configure a VPN routing and forwarding (VRF) instance on Router R4 to create a Layer 3 VPN.

```
[edit routing-instances coc-provider-vpn]
user@R4# set instance-type vrf
user@R4# set interface ge-2/0/0.0
user@R4# set interface ge-2/0/2.0
user@R4# set route-distinguisher 1.1.1.49:1
user@R4# set vrf-target target:300:1

[edit routing-instances coc-provider-vpn protocols bgp group toR3]
user@R4# set type external
user@R4# set family inet labeled-unicast per-prefix-label
user@R4# set neighbor 10.1.0.13 peer-as 200
```

Similarly, configure other VRF routing instances on R1, R6, R8, and R11.

## Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show policy-options**, **show routing-options**, **show protocols**, and **show routing-instances** commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
ge-2/0/0 {
  unit 0 {
    description toR3;
    family inet {
      address 10.1.0.14/30;
    }
    family iso;
    family mpls;
  }
}
ge-2/0/1 {
  unit 0 {
    description toR5;
    family inet {
      address 10.1.0.17/30;
    }
    family iso;
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 1.1.1.49/32;
    }
    family iso {
      address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00;
    }
  }
}
```

```

user@R4# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}

user@R4# show routing-options
router-id 1.1.1.49;
autonomous-system 300;
forwarding-table {
    export lb;
}

user@R4# show protocols
mpls {
    interface all;
}
ldp {
    track-igp-metric;
    interface ge-2/0/1.0;
    interface lo0.0;
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface ge-2/0/1.0 {
        level 2 metric 10;
    }
    interface lo0.0 {
        passive;
    }
}
bgp {
    group parent-vpn-peers {
        type internal;
        local-address 1.1.1.49;
        family inet-vpn {
            unicast;
        }
        neighbor 1.1.1.48;
        neighbor 1.1.1.62;
    }
}

user@R4# show routing-instances
coc-provider-vpn {
    instance-type vrf;
    interface ge-2/0/0.0;
    interface ge-2/0/2.0;
    route-distinguisher 1.1.1.49:1;
    vrf-target target:300:1;
    protocols {
        bgp {
            group toR3 {
                type external;
                family inet {
                    labeled-unicast {
                        per-prefix-label;
                    }
                }
            }
        }
    }
}

```

```

    }
    neighbor 10.1.0.13 {
        peer-as 200;
    }
}
}
}
}

```

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

Repeat the procedure for every router in this example, using the appropriate interface names and addresses for each router.

## Verification

Confirm that the configuration is working properly.

- [Enabling Protection on page 344](#)
- [Verifying Multipath Entries on page 344](#)
- [Verifying That Multipath Entries Have Different Weights on page 345](#)

### Enabling Protection

**Purpose** Enable protection on R4 to request protection for the link from R4 to R3.

**Action** 1. Add the **protection** statement at the [edit routing-instances *instance-name* protocols **bgp** group *group-name* family inet labeled-unicast] hierarchy level.

```

[edit routing-instances coc-provider-vpn protocols bgp group toR3]
user@R4# set family inet labeled-unicast protection

```

2. Verify and commit the configuration.

```

type external;
family inet {
    labeled-unicast {
        per-prefix-label;
        protection;
    }
}
neighbor 10.1.0.13 {
    peer-as 200;
}

```

### Verifying Multipath Entries

**Purpose** Verify that R4 has a multipath entry with two entries.

**Action** From operational mode on Router R4, run the **show route 1.1.1.52** command to check the route to R1.

```

user@R4> show route 1.1.1.52
#[Multipath/255] 00:02:44, metric 20
> to 10.1.0.13 via ge-2/0/0.0, Push 408592
  to 10.1.0.18 via ge-2/0/1.0, Push 299856, Push 299792(top)

```

### Verifying That Multipath Entries Have Different Weights

**Purpose** Verify that the two routes in the multipath entry have different weights, with the first entry having a weight of 0x1 and the second having a weight of 0x4000.

**Action** From operational mode on Router R4, run the **show route 1.1.1.52 detail** command to check the route to R1.

```
user@R4> show route 1.1.1.52 detail
#Multipath Preference: 255
  Next hop type: List, Next hop index: 1048609
  Address: 0x92f058c
  Next-hop reference count: 4
  Next hop: ELNH Address 0x92c48ac weight 0x1, selected
  equal-external-internal-type external
    Next hop type: Router, Next hop index: 1603
    Address: 0x92c48ac
    Next-hop reference count: 2
    Next hop: 10.1.0.13 via ge-2/0/0.0
    Label operation: Push 408592
    Label TTL action: prop-ttl
  Next hop: ELNH Address 0x92c548c weight 0x4000
  equal-external-internal-type internal
    Next hop type: Indirect
    Address: 0x92c548c
    Next-hop reference count: 3
    Protocol next hop: 1.1.1.62
    Push 299856
    Indirect next hop: 0x9380f40 1048608 INH Session ID: 0x10001a

    Next hop type: Router, Next hop index: 1586
    Address: 0x92c5440
    Next-hop reference count: 3
    Next hop: 10.1.0.18 via ge-2/0/1.0
    Label operation: Push 299856, Push 299792(top)
    Label TTL action: prop-ttl, prop-ttl(top)
  State: <ForwardingOnly Int Ext>
  Inactive reason: Forwarding use only
  Age: 3:38      Metric: 20
  Validation State: unverified
  Task: RT
  Announcement bits (1): 0-KRT
  AS path: 200 I
```

**Related Documentation**

- [Understanding Provider Edge Link Protection for BGP Labeled Unicast Paths on page 76](#)

### Example: Configuring Egress Protection for BGP Labeled Unicast

This example shows how to configure BGP labeled unicast protection that can be used in case of a PE failure in an Inter-AS Option C topology.

- [Requirements on page 346](#)
- [Overview on page 346](#)

- [Configuration on page 347](#)
- [Verification on page 357](#)

## Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 14.1 or later

## Overview

When network node or link failures occur, it takes some time to restore service using traditional routing table convergence. Local repair procedures can provide much faster restoration by establishing local protection as close to a failure as possible. Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or autonomous systems (ASs). If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

This example shows how to configure labeled-unicast egress protection in a Layer 3 VPN.

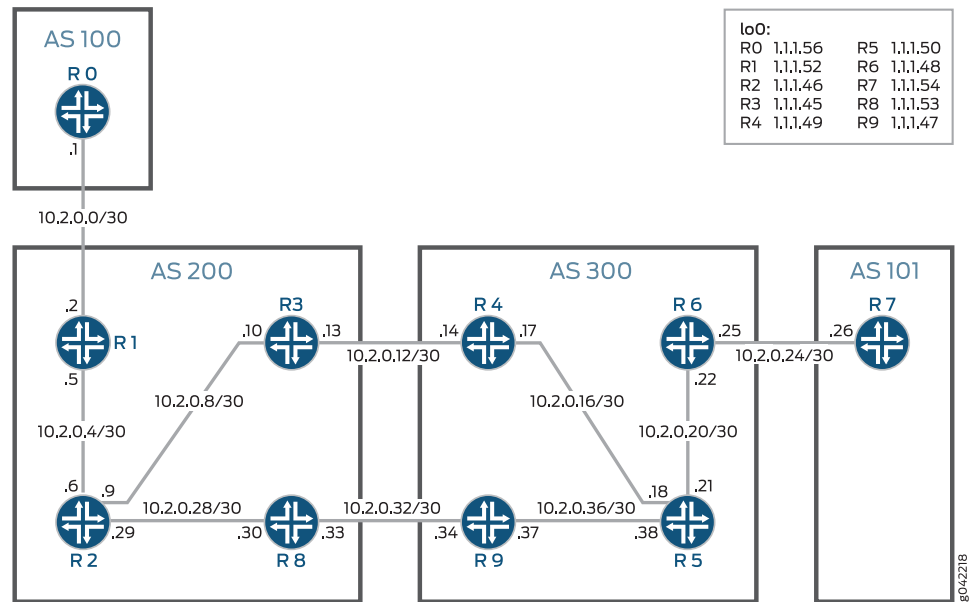
### Topology

---

In this example, an Inter-AS Option C topology is set up by configuring two customer edge (CE) devices and six service provider edge (PE) devices in four autonomous systems. The CE devices are configured in AS100 and AS101. The PE devices are configured in AS200 and AS300.

[Figure 41 on page 347](#) shows the topology used in this example.

Figure 41: Egress Protection in a Layer 3 VPN



The aim of this example is to protect PE Router R4. Egress protection is configured on Router R4 and Router R9 so that the traffic can be routed through the backup link (R9 to R8) when Router R4 (or the link from R5 to R4) goes down. In this example, Router R4 is the protected router, Router R9 is the protector router, and Router R5 is the point of local repair (PLR).

## Configuration

|                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>CLI Quick Configuration</b> | To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the <b>[edit]</b> hierarchy level.                                                                                                                                                                                                                                                                                                              |
| <b>Router R0</b>               | <pre> set interfaces ge-0/0/0 unit 0 description toR1 set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.1/30 set interfaces lo0 unit 0 family inet address 1.1.1.56/32 primary set routing-options router-id 1.1.1.56 set routing-options autonomous-system 100 set protocols ospf area 0.0.0.0 interface lo0.0 passive set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10 </pre>                                                                                                                                                                                       |
| <b>Router R1</b>               | <pre> set interfaces ge-0/0/0 unit 0 description toR0 set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.2/30 set interfaces ge-0/0/0 unit 0 family mpls set interfaces ge-0/0/1 unit 0 description toR2 set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.5/30 set interfaces ge-0/0/1 unit 0 family mpls set interfaces lo0 unit 0 family inet address 1.1.1.52/32 set routing-options router-id 1.1.1.52 set routing-options autonomous-system 200 set protocols mpls label-switched-path ToR3 to 1.1.1.45 set protocols mpls label-switched-path ToR8 to 1.1.1.53 </pre> |

```

set protocols mpls interface all
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.52
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers neighbor 1.1.1.45
set protocols bgp group parent-vpn-peers neighbor 1.1.1.53
set protocols bgp group toR6 type external
set protocols bgp group toR6 multihop ttl 10
set protocols bgp group toR6 local-address 1.1.1.52
set protocols bgp group toR6 family inet-vpn unicast
set protocols bgp group toR6 peer-as 300
set protocols bgp group toR6 neighbor 1.1.1.48
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement child_vpn_routes term 1 from protocol bgp
set policy-options policy-statement child_vpn_routes term 1 then accept
set policy-options policy-statement child_vpn_routes term 2 then reject
set policy-options policy-statement vpnexport term 1 from protocol ospf
set policy-options policy-statement vpnexport term 1 then community add test_comm
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
set policy-options policy-statement vpnimport term 1 from protocol bgp
set policy-options policy-statement vpnimport term 1 from community test_comm
set policy-options policy-statement vpnimport term 1 then accept
set policy-options policy-statement vpnimport term 2 then reject
set policy-options community text_comm members target:1:200
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-0/0/0.0
set routing-instances customer-provider-vpn route-distinguisher 1.1.1.45:1
set routing-instances customer-provider-vpn vrf-import vpnimport
set routing-instances customer-provider-vpn vrf-export vpnexport
set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
  ge-0/0/0.0

```

**Router R2**

```

set interfaces ge-0/0/0 unit 0 description toR3
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.9/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR1
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toR8
set interfaces ge-0/0/2 unit 0 family inet address 10.2.0.29/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.46/32
set routing-options router-id 1.1.1.46
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10

```

```

set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface lo0.0

```

```

Router R3
set interfaces ge-0/0/0 unit 0 description toR2
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.10/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR4
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.13/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.45/32
set routing-options router-id 1.1.1.45
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR1 to 1.1.1.52
set protocols mpls interface all
set protocols bgp group toR4 type external
set protocols bgp group toR4 family inet unicast
set protocols bgp group toR4 family inet labeled-unicast rib inet.3
set protocols bgp group toR4 export send-pe
set protocols bgp group toR4 neighbor 10.2.0.14 peer-as 300
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.45
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 1.1.1.52
set protocols bgp group parent-vpn-peers neighbor 1.1.1.53
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 1.1.1.52/32 exact
set policy-options policy-statement send-pe then accept

```

```

Router R4
set interfaces ge-0/0/0 unit 0 description toR5
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.17/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR3
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.14/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.49/32
set interfaces lo0 unit 0 family iso address
    47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
set routing-options router-id 1.1.1.49
set routing-options autonomous-system 300
set protocols mpls traffic-engineering bgp-igp-both-ribs

```

```

set protocols mpls label-switched-path ToR6 to 1.1.1.48
set protocols mpls interface all
set protocols mpls interface fxp.0 disable
set protocols mpls egress-protection context-identifier 100.1.1.1 primary
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.49
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-protection
context-identifier 100.1.1.1
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 1.1.1.48
set protocols bgp group parent-vpn-peers neighbor 1.1.1.47
set protocols bgp group toR3 type external
set protocols bgp group toR3 family inet labeled-unicast rib inet.3
set protocols bgp group toR3 export send-pe
set protocols bgp group toR3 peer-as 200
set protocols bgp group toR3 neighbor 10.2.0.13
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/0.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 1.1.1.48/32 exact
set policy-options policy-statement send-pe then accept

```

**Router R5**

```

set interfaces ge-0/0/0 unit 0 description toR4
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.18/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR6
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.21/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toR9
set interfaces ge-0/0/2 unit 0 family inet address 10.2.0.38/30
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.50/32
set interfaces lo0 unit 0 family iso address
47.0005.80ff.f800.0000.0108.0001.0102.5507.2050.00
set routing-options router-id 1.1.1.50
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols isis backup-spf-options per-prefix-calculation
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all node-link-protection
set protocols isis interface fxp0.0 disable
set protocols isis interface ge-0/0/0.0 link-protection
set protocols isis interface ge-0/0/0.0 level 2 metric 10
set protocols isis interface ge-0/0/1.0 link-protection

```

```

set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface ge-0/0/2.0 link-protection
set protocols isis interface ge-0/0/2.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable

```

```

Router R6
set interfaces ge-0/0/0 unit 0 description toR7
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.25/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR5
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.22/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.48/32
set interfaces lo0 unit 0 family iso address
    47.0005.80ff.f800.0000.0108.0001.0102.5507.2048.00
set routing-options router-id 1.1.1.48
set routing-options autonomous-system 300
set protocols mpls label-switched-path ToR4 to 1.1.1.49
set protocols mpls label-switched-path ToR9 to 1.1.1.47
set protocols mpls interface all
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.48
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers neighbor 1.1.1.49
set protocols bgp group parent-vpn-peers neighbor 1.1.1.47
set protocols bgp group toR1 type external
set protocols bgp group toR1 multihop ttl 10
set protocols bgp group toR1 local-address 1.1.1.48
set protocols bgp group toR1 family inet-vpn unicast
set protocols bgp group toR1 peers-as 200
set protocols bgp group toR1 neighbor 1.1.1.52
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement child-vpn-routes term 1 from protocol bgp
set policy-options policy-statement child-vpn-routes term 1 then accept
set policy-options policy-statement child-vpn-routes term 2 then reject
set policy-options policy-statement vpnexport term 1 from protocol ospf
set policy-options policy-statement vpnexport term 1 then community add test_comm
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
set policy-options policy-statement vpnimport term 1 from protocol bgp
set policy-options policy-statement vpnimport term 1 from community test_comm
set policy-options policy-statement vpnimport term 1 then accept
set policy-options policy-statement vpnimport term 2 then reject
set policy-options community test_comm members target:1:300
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-0/0/0.0

```

```

set routing-instances customer-provider-vpn route-distinguisher 1.1.1.49:1
set routing-instances customer-provider-vpn vrf-import vpnimport
set routing-instances customer-provider-vpn vrf-export vpnexport
set routing-instances customer-provider-vpn vrf-target target:300:1
set routing-instances customer-provider-vpn protocols ospf export child-vpn-routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
  ge-0/0/0.0

```

**Router R7**

```

set interfaces ge-0/0/0 unit 0 description toR6
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.26/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.54/32 primary
set routing-options router-id 1.1.1.54
set routing-options autonomous-system 101
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10

```

**Router R8**

```

set interfaces ge-0/0/0 unit 0 description toR9
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.33/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR2
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.30/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.53/32
set routing-options router-id 1.1.1.53
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR1 to 1.1.1.52
set protocols mpls interface all
set protocols bgp group toR9 type external
set protocols bgp group toR9 family inet unicast
set protocols bgp group toR9 family inet labeled-unicast rib inet.3
set protocols bgp group toR9 export send-pe
set protocols bgp group toR9 neighbor 10.2.0.34 peer-as 300
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.53
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 1.1.1.52
set protocols bgp group parent-vpn-peers neighbor 1.1.1.45
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 1.1.1.52/32 exact
set policy-options policy-statement send-pe then accept

```

**Router R9**

```

set interfaces ge-0/0/0 unit 0 description toR8

```

```

set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.34/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR5
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.37/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.47/32
set interfaces lo0 unit 0 family iso address
    47.0005.80ff.f800.0000.0108.0001.0102.5507.2062.00
set routing-options router-id 1.1.1.47
set routing-options autonomous-system 300
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR6 to 1.1.1.48
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 100.1.1.1 protector
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 1.1.1.47
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-protection
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 1.1.1.48
set protocols bgp group parent-vpn-peers neighbor 1.1.1.49
set protocols bgp group toR8 type external
set protocols bgp group toR8 family inet labeled-unicast rib inet.3
set protocols bgp group toR8 export send-pe
set protocols bgp group toR8 neighbor 10.2.0.33 peer-as 200
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 1.1.1.48/32 exact
set policy-options policy-statement send-pe then accept

```

### Configuring Egress Protection in Layer 3 VPNs

#### Step-by-Step Procedure

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

To configure labeled unicast egress protection:

1. Configure the interfaces on each router, for example:

```

[edit interfaces]
user@R4# set ge-0/0/0 unit 0 description toR5
user@R4# set ge-0/0/0 unit 0 family inet address 10.2.0.17/30
user@R4# set ge-0/0/0 unit 0 family iso
user@R4# set ge-0/0/0 unit 0 family mpls

user@R4# set ge-0/0/1 unit 0 description toR3
user@R4# set ge-0/0/1 unit 0 family inet address 10.2.0.14/30

```

```

user@R4# set ge-0/0/1 unit 0 family mpls
user@R4# set lo0 unit 0 family inet address 1.1.1.49/32
user@R4# set lo0 unit 0 family iso address
47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00

```

2. Configure the router ID and autonomous system (AS) number for each router, for example:

```

[edit routing-options]
user@R4# set router-id 1.1.1.49
user@R4# set autonomous-system 300

```

In this example, the router ID is chosen to be identical to the loopback address configured on the router.

3. Configure the protocols on each router, for example:

```

[edit protocols]
user@R4# set protocols mpls traffic-engineering bgp-igp-both-ribs
user@R4# set protocols mpls label-switched-path ToR6 to 1.1.1.48
user@R4# set protocols mpls interface all
user@R4# set protocols mpls interface fxp.0 disable
user@R4# set protocols bgp group parent-vpn-peers type internal
user@R4# set protocols bgp group parent-vpn-peers local-address 1.1.1.49
user@R4# set protocols bgp group parent-vpn-peers family inet unicast
user@R4# set protocols bgp group parent-vpn-peers family inet labeled-unicast
rib inet.3
user@R4# set protocols bgp group parent-vpn-peers export next-hop-self
user@R4# set protocols bgp group parent-vpn-peers neighbor 1.1.1.48
user@R4# set protocols bgp group parent-vpn-peers neighbor 1.1.1.47
user@R4# set protocols bgp group toR3 type external
user@R4# set protocols bgp group toR3 family inet labeled-unicast rib inet.3
user@R4# set protocols bgp group toR3 export send-pe
user@R4# set protocols bgp group toR3 peer-as 200
user@R4# set protocols bgp group toR3 neighbor 10.2.0.13
user@R4# set protocols isis level 1 disable
user@R4# set protocols isis level 2 wide-metrics-only
user@R4# set protocols isis interface ge-0/0/0.0 level 2 metric 10
user@R4# set protocols isis interface lo0.0 passive
user@R4# set protocols ldp interface ge-0/0/0.0
user@R4# set protocols ldp interface ge-0/0/1.0
user@R4# set protocols ldp interface lo0.0

```

4. Configure routing policies on all PE routers and AS border routers (Routers R1, R3, R4, R6, R8, and R9), for example:

```

user@R4# set policy-options policy-statement next-hop-self term 1 then
next-hop-self
user@R4# set policy-options policy-statement send-pe from route-filter 1.1.1.48/32
exact
user@R4# set policy-options policy-statement send-pe then accept

```

5. Configure the VPN routing instance on Routers R1 and R6.

```

user@R1# set routing-instances customer-provider-vpn instance-type vrf
user@R1# set routing-instances customer-provider-vpn interface ge-0/0/0.0
user@R1# set routing-instances customer-provider-vpn route-distinguisher 1.1.1.45:1
user@R1# set routing-instances customer-provider-vpn vrf-import vpnimport

```

```

user@R1# set routing-instances customer-provider-vpn vrf-export vpnexport
user@R1# set routing-instances customer-provider-vpn vrf-target target:200:1
user@R1# set routing-instances customer-provider-vpn protocols ospf export
  child_vpn_routes
user@R1# set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0
  interface ge-0/0/0.0

```

and

```

user@R6# set routing-instances customer-provider-vpn instance-type vrf
user@R6# set routing-instances customer-provider-vpn interface ge-0/0/0.0
user@R6# set routing-instances customer-provider-vpn route-distinguisher 1.1.1.49:1
user@R6# set routing-instances customer-provider-vpn vrf-import vpnimport
user@R6# set routing-instances customer-provider-vpn vrf-export vpnexport
user@R6# set routing-instances customer-provider-vpn vrf-target target:300:1
user@R6# set routing-instances customer-provider-vpn protocols ospf export
  child-vpn-routes
user@R6# set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0
  interface ge-0/0/0.0

```

6. Configure egress protection for Router R4, setting Router R4 as the protected router and Router R9 as the protector.

```

user@R4# set protocols mpls egress-protection context-identifier 100.1.1.1 primary
user@R4# set protocols bgp group parent-vpn-peers family inet labeled-unicast
  egress-protection context-identifier 100.1.1.1

```

and

```

user@R9# set protocols mpls egress-protection context-identifier 100.1.1.1 protector
user@R9# set protocols bgp group parent-vpn-peers family inet labeled-unicast
  egress-protection

```

## Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show routing-options**, **show protocols**, **show policy-options** (if applicable), and **show routing-instances** (if applicable) commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@R4# show interfaces
ge-0/0/0 {
  unit 0 {
    description toR5;
    family inet {
      address 10.2.0.17/30;
    }
    family iso;
    family mpls;
  }
}
ge-0/0/1 {
  unit 0 {
    description toR3;
    family inet {
      address 10.2.0.14/30;
    }
  }
}

```

```

        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 1.1.1.49/32;
        }
        family iso {
            address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00;
        }
    }
}

```

```

user@R4# show routing-options
router-id 1.1.1.49;
autonomous-system 300;

```

```

user@R4# show protocols
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path ToR6 {
        to 1.1.1.48;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
    egress-protection {
        context-identifier 100.1.1.1 {
            primary;
        }
    }
}
bgp {
    group parent-vpn-peers {
        type internal;
        local-address 1.1.1.49;
        family inet {
            unicast;
            labeled-unicast {
                rib {
                    inet.3;
                }
                egress-protection {
                    context-identifier {
                        100.1.1.1;
                    }
                }
            }
        }
    }
    export next-hop-self;
    neighbor 1.1.1.48;
    neighbor 1.1.1.47;
}
group toR3 {
    type external;
    family inet {
        unicast;
        labeled-unicast {
            rib {

```

```

        inet.3;
    }
}
}
export send-pe;
peer-as 200;
neighbor 10.2.0.13;
}
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface ge-0/0/0.0 {
        level 2 metric 10;
    }
    interface lo0.0 {
        passive;
    }
}
ldp {
    interface ge-0/0/0.0;
    interface ge-0/0/1.0;
    interface lo0.0;
}

user@R4# show policy-options
policy-statement next-hop-self {
    term 1 {
        then {
            next-hop self;
        }
    }
}
policy-statement send-pe {
    from {
        route-filter 1.1.1.48/32 exact;
    }
    then accept;
}

```

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

Repeat the procedure for every router in this example, using the appropriate interface names and addresses for each router.

## Verification

- [Verifying That Egress Protection Is Enabled on page 357](#)
- [Verifying the State of the Protected ASBR as 'primary' on page 358](#)
- [Verifying the State of the Protector ASBR as 'protector' on page 358](#)

### Verifying That Egress Protection Is Enabled

**Purpose** Verify that egress protection is enabled on the protected router, Router R4.

**Action** Run **show bgp neighbor** on Router R4 to verify that egress protection is enabled.

```
user@R4> show bgp neighbor
Peer: 1.1.1.47+45824 AS 300    Local: 1.1.1.49+27630 AS 300
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm    Last Event: RecvKeepAlive
  Last Error: None
  Export: [ next-hop-self ]
  Options: <Preference LocalAddress AddressFamily Refresh>
  Address families configured: inet-unicast inet-labeled-unicast
  Local Address: 1.1.1.49 Holdtime: 90 Preference: 170
NLRI configured with egress-protection: inet-labeled-unicast
Egress-protection NLRI inet-labeled-unicast context-identifier: 100.1.1.1
  Number of flaps: 0
  ...
```

#### Verifying the State of the Protected ASBR as 'primary'

**Purpose** Verify that the state of the protected AS border router, Router R4, is 'primary'.

**Action** Run **show mpls context-identifier** on Router R4.

```
user@R4> show mpls context-identifier
ID          Type      Metric    ContextTable
100.1.1.1    primary    1
Total 1, Primary 1, Protector 0
```

#### Verifying the State of the Protector ASBR as 'protector'

**Purpose** Verify that the state of the protector AS border router, Router R9, is 'protector'.

**Action** Run **show mpls context-identifier** on Router R9.

```
user@R9> show mpls context-identifier
ID          Type      Metric    ContextTable
100.1.1.1    protector  16777215  __100.1.1.1__.mpls.0
Total 1, Primary 0, Protector 1
```

- Related Documentation**
- [Egress Protection for BGP Labeled Unicast on page 77](#)
  - [Configuring Egress Protection for BGP Labeled Unicast on page 79](#)
  - [egress-protection \(BGP\) on page 524](#)

## Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains

This example shows how to configure Junos OS to tunnel IPv6 over a Layer 3 VPN IPv4 network. Internal BGP (IBGP) is used between the customer edge (CE) and provider edge (PE) devices, as described in Internet draft draft-marques-ppvpn-ibgp-version.txt,

*RFC2547bis networks using internal BGP as PE-CE protocol*, instead of the more typical external BGP (EBGP) PE-CE connections.

- [Requirements on page 359](#)
- [Overview on page 359](#)
- [Configuration on page 360](#)
- [Verification on page 365](#)

## Requirements

No special configuration beyond device initialization is required before you configure this example.

All PE routers participating in a Layer 3 VPN with the **independent-domain** statement in its configuration must be running Junos OS Release 6.3 or later.

## Overview

This example shows one method of enabling a router to participate in a customer VPN autonomous-system (AS) domain and to transparently exchange routing information through a Layer 3 VPN without the customer network attributes being visible to the carrier network, and without the carrier network attributes being visible to the customer network.

As an added requirement, the customer network in this example is based on IPv6, while the provider network uses IPv4.

The **independent-domain** feature is useful when customer route attributes need to be transparently forwarded across the VPN network without even the service-provider (SP) AS path appearing in the routes. In a typical Layer 3 VPN, the route attributes such as the originator ID, cluster list, route metric, and AS path are not transparent from one CE device to another CE device.

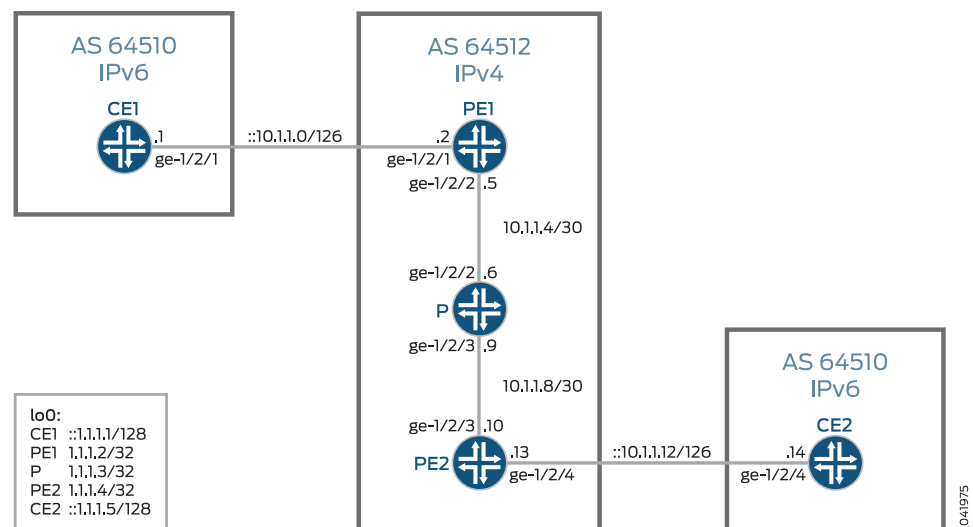
For example, suppose you have a customer VRF whose AS is 1. The customer advertises routes to you through BGP (either IBGP or EBGP). Your core network (the primary routing instance) uses AS 3. Without **independent-domain** configured, if the customer advertises 10.0.0.0/24 to you through BGP, the prefix contains the customer's AS 1 in the AS path. To transport the advertisement across the core to the other PE devices, your core AS 3 is added to the AS path by multiprotocol BGP (MP-BGP). The AS path is now 3 1. When the prefix is advertised out of the core back into the Layer 3 VPN at a remote PE device, the Layer 3 VPN AS 1 is added again, making the AS Path 1 3 1, which is an AS loop. The **independent-domain** statement ensures that only the ASs in the routing-instance are checked during loop detection, and the main, primary routing instances (your core's AS 3) is not considered. This is done by using the attribute 128 (attribute set), which is an optional transitive attribute. The attribute set hides the route's AS path, local preference, and so on, so that those do not appear during the loop check.



**NOTE:** In Junos OS 10.4 and later, you can specify the `no-attrset` option of `independent-domain` so that instead of using attribute 128 (attribute set), Junos OS simply does loop checking on routing-instance ASs without considering your core's AS used in MP-BGP. This is useful if you are using the `local-as` feature, and you only want to configure independent domains to maintain the independence of local ASs in the routing instance, and perform BGP loop detection only for the specified local ASs in the routing instance. In this case, you can disable the attribute set message.

Figure 42 on page 360 shows the sample network.

**Figure 42: Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains**



“CLI Quick Configuration” on page 360 shows the configuration for all of the devices in Figure 42 on page 360.

The section “Configuring Device PE1” on page 362 describes the steps on Device PE1.

## Configuration

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

**Device CE1**

```

set interfaces ge-1/2/1 unit 0 family inet6 address ::10.1.1.1/126
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet6 address ::1.1.1.1/128
set protocols bgp group toPE1 type internal
set protocols bgp group toPE1 family inet6 unicast
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 neighbor ::10.1.1.2
set policy-options policy-statement send-direct from protocol direct

```

```

set policy-options policy-statement send-direct then accept
set routing-options router-id 1.1.1.1
set routing-options autonomous-system 64510

Device CE2
set interfaces ge-1/2/4 unit 0 family inet6 address ::10.1.1.14/126
set interfaces ge-1/2/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet6 address ::1.1.1.5/128
set protocols bgp group toPE2 type internal
set protocols bgp group toPE2 family inet6 unicast
set protocols bgp group toPE2 export send-direct
set protocols bgp group toPE2 neighbor ::10.1.1.13
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 1.1.1.5
set routing-options autonomous-system 64510

Device PE1
set interfaces ge-1/2/1 unit 0 family inet6 address ::10.1.1.2/126
set interfaces ge-1/2/2 unit 0 family inet address 10.1.1.5/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.2/32
set protocols mpls ipv6-tunneling
set protocols mpls interface ge-1/2/2.0
set protocols bgp group toPE2 type internal
set protocols bgp group toPE2 local-address 1.1.1.2
set protocols bgp group toPE2 family inet6-vpn unicast
set protocols bgp group toPE2 neighbor 1.1.1.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-instances red instance-type vrf
set routing-instances red interface ge-1/2/1.0
set routing-instances red route-distinguisher 64512:1
set routing-instances red vrf-target target:64512:1
set routing-instances red routing-options router-id 1.1.1.2
set routing-instances red routing-options autonomous-system 64510
set routing-instances red routing-options autonomous-system independent-domain
set routing-instances red protocols bgp group toCE1 type internal
set routing-instances red protocols bgp group toCE1 family inet6 unicast
set routing-instances red protocols bgp group toCE1 neighbor ::10.1.1.1
set routing-options router-id 1.1.1.2
set routing-options autonomous-system 64512

Device P
set interfaces ge-1/2/2 unit 0 family inet address 10.1.1.6/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/3 unit 0 family inet address 10.1.1.9/30
set interfaces ge-1/2/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 1.1.1.3/32
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface all
set protocols ldp interface all
set routing-options router-id 1.1.1.3

Device PE2
set interfaces ge-1/2/3 unit 0 family inet address 10.1.1.10/30

```

```

set interfaces ge-1/2/3 unit 0 family mpls
set interfaces ge-1/2/4 unit 0 family inet6 address ::10.1.1.13/126
set interfaces lo0 unit 0 family inet address 1.1.1.4/32
set protocols mpls ipv6-tunneling
set protocols mpls interface ge-1/2/3.0
set protocols bgp group toPE1 type internal
set protocols bgp group toPE1 local-address 1.1.1.4
set protocols bgp group toPE1 family inet6-vpn unicast
set protocols bgp group toPE1 neighbor 1.1.1.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/3.0
set protocols ldp interface ge-1/2/3.0
set protocols ldp interface lo0.0
set routing-instances red instance-type vrf
set routing-instances red interface ge-1/2/4.0
set routing-instances red route-distinguisher 64512:1
set routing-instances red vrf-target target:64512:1
set routing-instances red routing-options router-id 1.1.1.4
set routing-instances red routing-options autonomous-system 64510
set routing-instances red routing-options autonomous-system independent-domain
set routing-instances red protocols bgp group toCE2 type internal
set routing-instances red protocols bgp group toCE2 family inet6 unicast
set routing-instances red protocols bgp group toCE2 neighbor ::10.1.1.14
set routing-options router-id 1.1.1.4
set routing-options autonomous-system 64512

```

### Configuring Device PE1

#### Step-by-Step Procedure

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

To configure Device PE1:

1. Configure the interfaces.

```

[edit interfaces]
user@PE1# set ge-1/2/1 unit 0 family inet6 address ::10.1.1.2/126

user@PE1# set ge-1/2/2 unit 0 family inet address 10.1.1.5/30
user@PE1# set ge-1/2/2 unit 0 family mpls

user@PE1# set lo0 unit 0 family inet address 1.1.1.2/32

```

2. Configure MPLS on the interfaces.

```

[edit protocols mpls]
user@PE1# set ipv6-tunneling
user@PE1# set interface ge-1/2/2.0

```

3. Configure BGP.

```

[edit protocols bgp group toPE2]
user@PE1# set type internal
user@PE1# set local-address 1.1.1.2
user@PE1# set family inet6-vpn unicast

```

- ```

user@PE1# set neighbor 1.1.1.4

```
4. Configure an interior gateway protocol (IGP).
 

```

[edit protocols ospf area 0.0.0.0]
user@PE1# set interface lo0.0 passive
user@PE1# set interface ge-1/2/2.0

```
  5. Configure a signaling protocol.
 

```

[edit protocols]
user@PE1# set ldp interface ge-1/2/2.0
user@PE1# set ldp interface lo0.0

```
  6. Configure the routing instance.
 

```

[edit routing-instances red]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/1.0
user@PE1# set route-distinguisher 64512:1
user@PE1# set vrf-target target:64512:1
user@PE1# set routing-options router-id 1.1.1.2
user@PE1# set protocols bgp group toCE1 type internal
user@PE1# set protocols bgp group toCE1 family inet6 unicast
user@PE1# set protocols bgp group toCE1 neighbor ::10.1.1.1

```
  7. In the routing instance, include the AS number of the customer network, and include the **independent-domain** statement.
 

```

[edit routing-instances red routing-options]
user@PE1# set autonomous-system 64510
user@PE1# set autonomous-system independent-domain

```
  8. In the main instance, configure the router ID and the provider AS number.
 

```

[edit routing-options]
user@PE1# set router-id 1.1.1.2
user@PE1# set autonomous-system 64512

```

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

```

user@PE1# show interfaces
interfaces {
  ge-1/2/1 {
    unit 0 {
      family inet6 {
        address ::10.1.1.2/126;
      }
    }
  }
  ge-1/2/2 {
    unit 0 {
      family inet {
        address 10.1.1.5/30;
      }
    }
  }
}

```

```
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 1.1.1.2/32;
        }
    }
}
}

user@PE1# show protocols
mpls {
    ipv6-tunneling;
    interface ge-1/2/2.0;
}
bgp {
    group toPE2 {
        type internal;
        local-address 1.1.1.2;
        family inet6-vpn {
            unicast;
        }
        neighbor 1.1.1.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-1/2/2.0;
    }
}
ldp {
    interface ge-1/2/2.0;
    interface lo0.0;
}

user@PE1# show routing-instances
red {
    instance-type vrf;
    interface ge-1/2/1.0;
    route-distinguisher 64512:1;
    vrf-target target:64512:1;
    routing-options {
        router-id 1.1.1.2;
        autonomous-system 64510 independent-domain;
    }
    protocols {
        bgp {
            group toCE1 {
                type internal;
                family inet6 {
                    unicast;
                }
            }
        }
    }
}
```

```

        neighbor ::10.1.1.1;
    }
}
}

user@PE1# show routing-options
router-id 1.1.1.2;
autonomous-system 64512;

```

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

## Verification

Confirm that the configuration is working properly.

- [Verifying That the CE Devices Have Connectivity on page 365](#)
- [Checking the AS Paths on page 365](#)

### Verifying That the CE Devices Have Connectivity

<b>Purpose</b>	Make sure that the tunnel is operating.
<b>Action</b>	<p>From operational mode, enter the <b>ping</b> command.</p> <pre> user@CE1&gt; ping ::1.1.1.5  PING6(56=40+8+8 bytes) ::10.1.1.1 --&gt; ::1.1.1.5 16 bytes from ::1.1.1.5, icmp_seq=0 hlim=63 time=1.943 ms 16 bytes from ::1.1.1.5, icmp_seq=1 hlim=63 time=1.587 ms ^C --- ::1.1.1.5 ping6 statistics --- 2 packets transmitted, 2 packets received, 0% packet loss round-trip min/avg/max/std-dev = 1.587/1.765/1.943/0.178 ms  user@CE2&gt; ping ::1.1.1.1 PING6(56=40+8+8 bytes) ::10.1.1.14 --&gt; ::1.1.1.1 16 bytes from ::1.1.1.1, icmp_seq=0 hlim=63 time=2.097 ms 16 bytes from ::1.1.1.1, icmp_seq=1 hlim=63 time=1.610 ms ^C --- ::1.1.1.1 ping6 statistics --- 2 packets transmitted, 2 packets received, 0% packet loss round-trip min/avg/max/std-dev = 1.610/1.853/2.097/0.244 ms </pre>
<b>Meaning</b>	The IPv6 CE devices can communicate over the core IPv4 network.

### Checking the AS Paths

<b>Purpose</b>	Make sure that the provider AS number does not appear in the CE device routing tables.
<b>Action</b>	<p>From operational mode, enter the <b>show route protocol bgp detail</b> command.</p> <pre> user@CE1&gt; show route protocol bgp detail  inet6.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden) ::1.1.1.5/128 (1 entry, 1 announced) </pre>

```

*BGP      Preference: 170/-101
          Next hop type: Indirect
          Address: 0x9514354
          Next-hop reference count: 6
          Source: ::10.1.1.2
          Next hop type: Router, Next hop index: 924
          Next hop: ::10.1.1.2 via ge-1/2/1.0, selected
          Session Id: 0x500001
          Protocol next hop: ::10.1.1.2
          Indirect next hop: 0x971c000 262147 INH Session ID: 0x500002
          State: <Active Int Ext>
          Local AS: 64510 Peer AS: 64510
          Age: 50:58      Metric2: 0
          Validation State: unverified
          Task: BGP_64510::10.1.1.2+45824
          Announcement bits (2): 0-KRT 2-Resolve tree 2
AS path: I
          Accepted
          Localpref: 100
          Router ID: 1.1.1.2

::10.1.1.12/126 (1 entry, 1 announced)
*BGP      Preference: 170/-101
          Next hop type: Indirect
          Address: 0x9514354
          Next-hop reference count: 6
          Source: ::10.1.1.2
          Next hop type: Router, Next hop index: 924
          Next hop: ::10.1.1.2 via ge-1/2/1.0, selected
          Session Id: 0x500001
          Protocol next hop: ::10.1.1.2
          Indirect next hop: 0x971c000 262147 INH Session ID: 0x500002
          State: <Active Int Ext>
          Local AS: 64510 Peer AS: 64510
          Age: 50:58      Metric2: 0
          Validation State: unverified
          Task: BGP_64510::10.1.1.2+45824
          Announcement bits (2): 0-KRT 2-Resolve tree 2
AS path: I
          Accepted
          Localpref: 100
          Router ID: 1.1.1.2

```

user@CE2> show route protocol bgp detail

```

inet6.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
::1.1.1.1/128 (1 entry, 1 announced)
*BGP      Preference: 170/-101
          Next hop type: Indirect
          Address: 0x9514354
          Next-hop reference count: 6
          Source: ::10.1.1.13
          Next hop type: Router, Next hop index: 914
          Next hop: ::10.1.1.13 via ge-1/2/4.0, selected
          Session Id: 0x400001
          Protocol next hop: ::10.1.1.13
          Indirect next hop: 0x971c000 262150 INH Session ID: 0x400002
          State: <Active Int Ext>
          Local AS: 64510 Peer AS: 64510
          Age: 50:41      Metric2: 0

```

```

Validation State: unverified
Task: BGP_64510::10.1.1.13+59329
Announcement bits (2): 0-KRT 2-Resolve tree 2
AS path: I
Accepted
Localpref: 100
Router ID: 1.1.1.4

::10.1.1.0/126 (1 entry, 1 announced)
*BGP Preference: 170/-101
Next hop type: Indirect
Address: 0x9514354
Next-hop reference count: 6
Source: ::10.1.1.13
Next hop type: Router, Next hop index: 914
Next hop: ::10.1.1.13 via ge-1/2/4.0, selected
Session Id: 0x400001
Protocol next hop: ::10.1.1.13
Indirect next hop: 0x971c000 262150 INH Session ID: 0x400002
State: <Active Int Ext>
Local AS: 64510 Peer AS: 64510
Age: 50:41 Metric2: 0
Validation State: unverified
Task: BGP_64510::10.1.1.13+59329
Announcement bits (2): 0-KRT 2-Resolve tree 2
AS path: I
Accepted
Localpref: 100
Router ID: 1.1.1.4

```

**Meaning** The output shows that for the BGP routes on the CE devices, the AS path attribute does not include the provider AS 64512.

**Related Documentation**

- *Configuring the Ingress Router for MPLS-Signaled LSPs*
- *Configuring the Intermediate and Egress Routers for MPLS-Signaled LSPs*
- *Minimum RSVP Configuration*



## CHAPTER 5

# Layer 3 VPN Internet Access Examples

- [Non-VRF Internet Access on page 369](#)
- [Distributed Internet Access on page 370](#)
- [Routing VPN and Internet Traffic Through Different Interfaces on page 371](#)
- [Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface on page 377](#)
- [Routing VPN and Internet Traffic Through the Same Interface Bidirectionally \(VPN Has Public Addresses\) on page 378](#)
- [Routing VPN and Internet Traffic Through the Same Interface Bidirectionally \(VPN Has Private Addresses\) on page 382](#)
- [Routing Internet Traffic Through a Separate NAT Device on page 386](#)
- [Centralized Internet Access on page 394](#)

## Non-VRF Internet Access

---

Junos OS supports Internet access from a Layer 3 virtual private network (VPN). This chapter provides examples that demonstrate how to configure a provider edge (PE) router to provide Internet access to customer edge (CE) routers in a VPN. The method you use depends on the needs and specifications of the individual network. To provide Internet access through a Layer 3 VPN, you need to configure policies on the PE router. You also need to configure the **next-table** statement at the **[edit routing-instances routing-instance-name routing-options static route]** hierarchy level. When configured, this statement can point a default route from the VPN table (routing instance) to the main routing table (default instance) inet.0. The main routing table stores all Internet routes and is where final route resolution occurs.

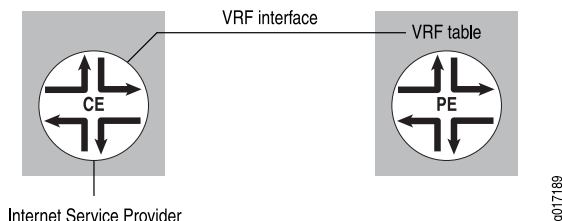
The following sections describe ways to provide Internet access to a CE router in a Layer 3 VPN without using the VPN routing and forwarding (VRF) interface. Because these methods effectively bypass the Layer 3 VPN, they are not discussed in detail.

- [CE Router Accesses Internet Independently of the PE Router on page 370](#)
- [PE Router Provides Layer 2 Internet Service on page 370](#)

## CE Router Accesses Internet Independently of the PE Router

In this configuration, the PE router does not provide the Internet access. The CE router sends Internet traffic either to another service provider, or to the same service provider but a different router. The PE router handles Layer 3 VPN traffic only (see [Figure 43 on page 370](#)).

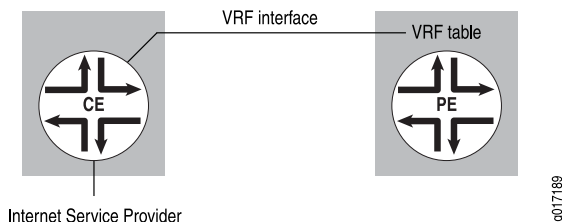
**Figure 43: PE Router Does Not Provide Internet Access**



## PE Router Provides Layer 2 Internet Service

In this configuration, the PE router acts as a Layer 2 device, providing a Layer 2 connection (such as circuit cross-connect [CCC]) to another router that has a full set of Internet routes. The CE router can use just one physical interface and two logical interfaces to the PE router, or it can use multiple physical interfaces to the PE router (see [Figure 44 on page 370](#)).

**Figure 44: PE Router Connects to a Router Connected to the Internet**



## Distributed Internet Access

In this scenario, the PE routers provide Internet access to the CE routers. In the examples that follow, it is assumed that the Internet routes (or defaults) are present in the inet.0 table of the PE routers that provide Internet access to selected CE routers.

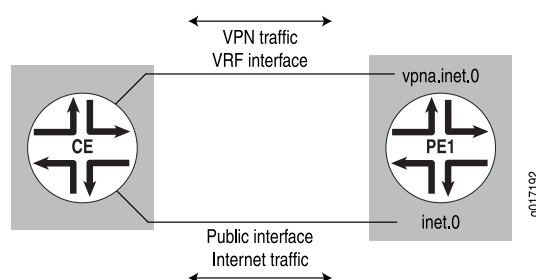
When accessing the Internet from a VPN, Network Address Translation (NAT) must be performed between the VPN's private addresses and the public addresses used on the Internet unless the VPN is using the public address space. This section includes several examples of how to provide Internet access for VPNs, most of which require that the CE routers perform the address translation. The [“Routing Internet Traffic Through a Separate NAT Device” on page 386](#) example, however, requires that the service provider supply the NAT functionality using a NAT device connected to the PE router.

In all of the examples, the VPN's public IP address pool (whose entries correspond to the translated private addresses) must be added to the inet.0 table and propagated to the Internet routers to receive reverse traffic from public destinations.

## Routing VPN and Internet Traffic Through Different Interfaces

In this example, VPN and Internet traffic are routed through different interfaces. The CE router sends the VPN traffic through the VPN interface and sends the Internet traffic through a separate interface that is part of the main routing table on Router PE1 (the CE router can use either one physical interface with two logical units or two physical interfaces). NAT also occurs on the CE router (see [Figure 45 on page 371](#)).

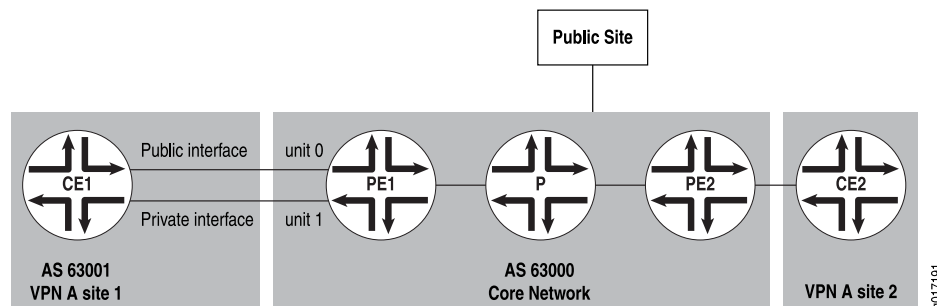
**Figure 45: Routing VPN and Internet Traffic Through Different Interfaces**



The PE router is configured to install and advertise the public IP address pool for the VPN to other core routers (for return traffic). The VPN traffic is routed normally.

[Figure 46 on page 371](#) illustrates the PE router's VPN configuration.

**Figure 46: Example of Internet Traffic Routed Through Separate Interfaces**



The configuration in this example has the following features:

- Router PE1 uses two logical interfaces to connect to Router CE1 using Frame Relay encapsulation.
- The routing protocol between Router PE1 and Router CE1 is EBGp.
- Router CE1's public IP address pool is 10.12.1.1 through 10.12.1.254 (10.12.1.0/24).
- The **next-hop-self** setting is derived from the **fix-nh policy** statement on Router PE1. PE routers are forced to use **next-hop-self** so that next-hop resolution is done only for the PE router's loopback address for non-VPN routes (by default, VPN–Internet Protocol version 4 [IPv4] routes are sent by means of **next-hop-self**).

You can configure Router CE1 with a static default route pointing to its public interface for everything else.

The following sections show how to route VPN and Internet traffic through different interfaces:

- [Configuring Interfaces on Router PE1 on page 372](#)
- [Configuring Routing Options on Router PE1 on page 372](#)
- [Configuring BGP, IS-IS, and LDP Protocols on Router PE1 on page 372](#)
- [Configuring a Routing Instance on Router PE1 on page 373](#)
- [Configuring Policy Options on Router PE1 on page 374](#)
- [Traffic Routed by Different Interfaces: Configuration Summarized by Router on page 375](#)

## Configuring Interfaces on Router PE1

Configure an interface to handle VPN traffic and an interface to handle Internet traffic:

```
[edit]
interfaces {
  t3-0/2/0 {
    dce;
    encapsulation frame-relay;
    unit 0 {
      description "to CE1 VPN interface";
      dlci 10;
      family inet {
        address 192.168.197.13/30;
      }
    }
    unit 1 {
      description "to CE1 public interface";
      dlci 20;
      family inet {
        address 192.168.198.201/30;
      }
    }
  }
}
```

## Configuring Routing Options on Router PE1

Configure a static route on Router PE1 to install a route to the CE router's public IP address pool in inet.0:

```
[edit]
routing-options {
  static {
    route 10.12.1.0/24 next-hop 192.168.198.202;
  }
}
```

## Configuring BGP, IS-IS, and LDP Protocols on Router PE1

Configure BGP on Router PE1 to allow non-VPN and VPN peering and to advertise the VPN's public IP address pool:

```
[edit]
```

```

protocols {
  bgp {
    group pe-pe {
      type internal;
      local-address 10.255.14.171;
      family inet {
        any;
      }
      family inet-vpn {
        any;
      }
      export [fix-nh redist-static];
      neighbor 10.255.14.177;
      neighbor 10.255.14.179;
    }
  }
}

```

Configure IS-IS on Router PE1 to allow access to internal routes:

```

[edit protocols]
isis {
  level 1 disable;
  interface so-0/0/0.0;
  interface lo0.0;
}

```

Configure LDP on Router PE1 to tunnel VPN routes:

```

[edit protocols]
ldp {
  interface so-0/0/0.0;
}

```

## Configuring a Routing Instance on Router PE1

Configure a routing instance on Router PE1:

```

[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
      bgp {
        group to-CE1 {
          peer-as 63001;
          neighbor 192.168.197.14;
        }
      }
    }
  }
}

```

## Configuring Policy Options on Router PE1

You need to configure policy options on Router PE1. The **fix-nh** policy statement sets **next-hop-self** for all non-VPN routes:

```
[edit]
policy-options {
  policy-statement fix-nh {
    then {
      next-hop self;
    }
  }
}
```

The **redist-static** policy statement advertises the VPN's public IP address pool:

```
[edit policy-options]
policy-statement redist-static {
  term a {
    from {
      protocol static;
      route-filter 10.12.1.0/24 exact;
    }
    then accept;
  }
  term b {
    then reject;
  }
}
```

Configure import and export policies for **vpna**:

```
[edit policy-options]
policy-statement vpna-import {
  term a {
    from {
      protocol bgp;
      community vpna-comm;
    }
    then accept;
  }
  term b {
    then reject;
  }
}
policy-statement vpna-export {
  term a {
    from protocol bgp;
    then {
      community add vpna-comm;
      accept;
    }
  }
  term b {
    then reject;
  }
}
```

```

}
community vpna-comm members target:63000:100;

```

## Traffic Routed by Different Interfaces: Configuration Summarized by Router

### Router PE1

<b>Interfaces</b>	<pre> interfaces {   t3-0/2/0 {     dce;     encapsulation frame-relay;     unit 0 {       description "to CE1 VPN interface";       dlci 10;       family inet {         address 192.168.197.13/30;       }     }     unit 1 {       description "to CE1 public interface";       dlci 20;       family inet {         address 192.168.198.201/30;       }     }   } } </pre>
<b>Routing Options</b>	<pre> routing-options {   static {     route 10.12.1.0/24 next-hop 192.168.198.202;   } } </pre>
<b>BGP Protocol</b>	<pre> protocols {   bgp {     group pe-pe {       type internal;       local-address 10.255.14.171;       family inet {         any;       }       family inet-vpn {         any;       }       export [ fix-nh redist-static];       neighbor 10.255.14.177;       neighbor 10.255.14.179;     }   } } </pre>
<b>IS-IS Protocol</b>	<pre> isis {   level 1 disable;   interface so-0/0/0.0; } </pre>

	<pre> interface lo0.0; } </pre>
<b>LDP Protocol</b>	<pre> ldp {   interface so-0/0/0.0; } </pre>
<b>Routing Instance</b>	<pre> routing-instances {   vpna {     instance-type vrf;     interface t3-0/2/0.0;     route-distinguisher 10.255.14.171:100;     vrf-import vpna-import;     vrf-export vpna-export;     protocols {       bgp {         group to-CE1 {           peer-as 63001;           neighbor 192.168.197.14;         }       }     }   } } </pre>
<b>Policy Options/Policy Statements</b>	<pre> policy-options {   policy-statement fix-nh {     then {       next-hop self;     }   }   policy-statement redist-static {     term a {       from {         protocol static;         route-filter 10.12.1.0/24 exact;       }       then accept;     }     term b {       then reject;     }   } } </pre>
<b>Import and Export Policies</b>	<pre> policy-statement vpna-import {   term a {     from {       protocol bgp;       community vpna-comm;     }     then accept;   }   term b {     then reject;   } } </pre>

```

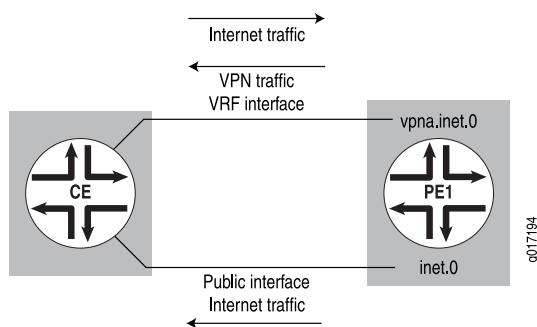
    }
  }
  policy-statement vpna-export {
    term a {
      from protocol bgp;
      then {
        community add vpna-comm;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  community vpna-comm members target:63000:100;

```

## Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface

In this example, the CE router sends VPN and Internet traffic through the same interface but receives return Internet traffic through a different interface. The PE router has a default route in the VRF table pointing to the main routing table inet.0. It routes the VPN public IP address pool (return Internet traffic) through a different interface in inet.0 (see [Figure 47 on page 377](#)). The CE router still performs NAT functions.

**Figure 47: VPN and Outgoing Internet Traffic Routed Through the Same Interface and Return Internet Traffic Routed Through a Different Interface**



The following section shows how to route VPN and outgoing Internet traffic through the same interface and routing return Internet traffic through a different interface:

- [Configuration for Router PE1 on page 377](#)

### Configuration for Router PE1

This example has the same configuration as Router PE1 in [“Routing VPN and Internet Traffic Through Different Interfaces” on page 371](#). It uses the topology shown in [Figure 46 on page 371](#). The default route to the VPN routing table is configured differently. At the `[edit routing-instances routing-instance-name routing-options]` hierarchy level, you configure a default static route that is installed in `vpna.inet.0` and points to `inet.0` for resolution:

```
[edit]
```

```

routing-instances {
  vpn {
    instance-type vrf;
    interface t3-0/2/0.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpn-import;
    vrf-export vpn-export;
    routing-options {
      static {
        route 0.0.0.0/0 next-table inet.0;
      }
    }
    protocols {
      bgp {
        group to-CE1 {
          peer-as 63001;
          neighbor 192.168.197.14;
        }
      }
    }
  }
}

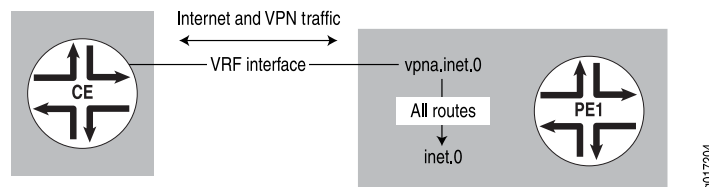
```

You also need to change the configuration of Router CE1 (from the configuration that works with the configuration for Router PE1 described in [“Routing VPN and Internet Traffic Through Different Interfaces” on page 371](#)) to account for the differences in the configuration of the PE routers.

## Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses)

This section shows how to configure a single logical interface to handle VPN and Internet traffic traveling both to and from the Internet and the CE router. This interface can handle both VPN and Internet traffic as long as there are no private addresses in the VPN. The VPN routes received from the CE router are added to the main routing table inet.0 by means of routing table groups. This allows the PE router to attract the return traffic from the Internet (see [Figure 48 on page 378](#)).

**Figure 48: Interface Configured to Carry Both Internet and VPN Traffic**



In this example, the CE router does not need to perform NAT, because all the VPN routes are public. The CE router has a single interface to the PE router, to which it advertises VPN routes. The PE router has a default route in the VRF table pointing to the main routing table inet.0. The PE router also imports VPN routes received from the CE router into inet.0 by means of routing table groups.

The following configuration for Router PE1 uses the same topology as in [“Routing VPN and Internet Traffic Through Different Interfaces” on page 371](#). This configuration uses a single logical interface (instead of two) between Router PE1 and Router CE1.

The following sections show how to route VPN and Internet traffic through the same interface bidirectionally (VPN has public addresses):

- [Configuring Routing Options on Router PE1 on page 379](#)
- [Configuring Routing Protocols on Router PE1 on page 379](#)
- [Configuring the Routing Instance on Router PE1 on page 380](#)
- [Traffic Routed Through the Same Interface Bidirectionally: Configuration Summarized by Router on page 381](#)

## Configuring Routing Options on Router PE1

Configure a routing table group definition for installing VPN routes in routing table groups `vpna.inet.0` and `inet.0`:

```
[edit]
routing-options {
  rib-groups {
    vpna-to-inet0 {
      import-rib [ vpna.inet.0 inet.0 ];
    }
  }
}
```

## Configuring Routing Protocols on Router PE1

Configure MPLS, BGP, IS-IS, and LDP protocols on Router PE1. This configuration does not include the **policy redist-static** statement at the **[edit protocols bgp group pe-pe]** hierarchy level. The VPN routes are sent directly to IBGP.

Configure BGP on Router PE1 to allow non-VPN and VPN peering, and to advertise the VPN's public IP address pool:

```
[edit]
protocols {
  mpls {
    interface so-0/0/0.0;
  }
  bgp {
    group pe-pe {
      type internal;
      local-address 10.255.14.171;
      family inet {
        any;
      }
      family inet-vpn {
        any;
      }
      export fix-nh;
      neighbor 10.255.14.177;
      neighbor 10.255.14.173;
    }
  }
}
```

```

    }
  }
  isis {
    level 1 disable;
    interface so-0/0/0.0;
    interface lo0.0;
  }
  ldp {
    interface so-0/0/0.0;
  }
}

```

## Configuring the Routing Instance on Router PE1

This section describes how to configure the routing instance on Router PE1. The static route defined in the **routing-options** statement directs Internet traffic from the CE router to the inet.0 routing table. The routing table group defined by the **rib-group vpna-to-inet0** statement adds the VPN routes to inet.0.

Configure the routing instance on Router PE1:

```

[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      static {
        route 0.0.0.0/0 next-table inet.0;
      }
    }
    protocols {
      bgp {
        group to-CE1 {
          family inet {
            unicast {
              rib-group vpna-to-inet0;
            }
          }
        }
        peer-as 63001;
        neighbor 192.168.197.14;
      }
    }
  }
}

```

You must configure Router CE1 to forward all traffic to Router PE1 using a default route. Alternatively, the default route can be advertised from Router PE1 to Router CE1 with EBGP.

## Traffic Routed Through the Same Interface Bidirectionally: Configuration Summarized by Router

### Router PE1

This example uses the same configuration as in [“Routing VPN and Internet Traffic Through Different Interfaces” on page 371](#). This configuration uses a single logical interface (instead of two) between Router PE1 and Router CE1.

Routing Options	<pre> routing-options {   rib-groups {     vpna-to-inet0 {       import-rib [ vpna.inet.0 inet.0 ];     }   } } </pre>
Routing Protocols	<pre> protocols {   mpls {     interface so-0/0/0.0;   }   bgp {     group pe-pe {       type internal;       local-address 10.255.14.171;       family inet {         any;       }       family inet-vpn {         any;       }       export fix-nh;       neighbor 10.255.14.177;       neighbor 10.255.14.173;     }   }   isis {     level 1 disable;     interface so-0/0/0.0;     interface lo0.0;   }   ldp {     interface so-0/0/0.0;   } } </pre>
Routing Instance	<pre> routing-instances {   vpna {     instance-type vrf;     interface t3-0/2/0.0;     route-distinguisher 10.255.14.171:100;     vrf-import vpna-import;     vrf-export vpna-export;     routing-options {       static {         route 0.0.0.0/0 next-table inet.0;       }     }   } } </pre>

```

    }
  }
  protocols {
    bgp {
      group to-CE1 {
        family inet {
          unicast {
            rib-group vpna-to-inet0;
          }
        }
      }
      peer-as 63001;
      neighbor 192.168.197.14;
    }
  }
}

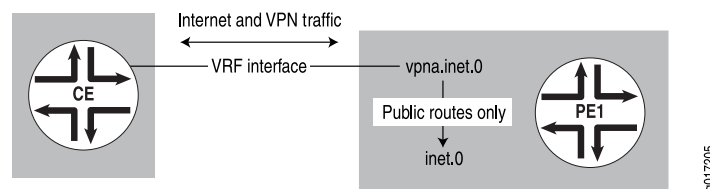
```

## Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Private Addresses)

The example in this section shows how to route VPN and Internet traffic through the same interface in both directions (from the CE router to the Internet and from the Internet to the CE router). The VPN in this example has private addresses. If you can configure EBGp on the CE router, you can configure a PE router using the configuration outlined in “Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses)” on page 378, even if the VPN has private addresses.

In the example described in this section, the CE router uses separate communities to advertise its VPN routes and public routes. The PE router selectively imports only the public routes into the inet.0 routing table. This configuration ensures that return traffic from the Internet uses the same interface between the PE and CE routers as that used by VPN traffic going out to public Internet addresses (see Figure 49 on page 382).

**Figure 49: VPN and Internet Traffic Routed Through the Same Interface**



In this example, the CE router has one interface and a BGP session with the PE router, and it tags VPN routes and Internet routes with different communities. The PE router has one interface, selectively imports routes for the VPN's public IP address pool into inet.0, and has a default route in the VRF routing table pointing to inet.0.

The following sections show how to route VPN and Internet traffic through the same interface bidirectionally (VPN has private addresses):

- [Configuring Routing Options for Router PE1 on page 383](#)
- [Configuring a Routing Instance for Router PE1 on page 383](#)

- [Configuring Policy Options for Router PE1 on page 384](#)
- [Traffic Routed by the Same Interface Bidirectionally \(VPN Has Private Addresses\): Configuration Summarized by Router on page 384](#)

## Configuring Routing Options for Router PE1

On Router PE1, configure a routing table group to install VPN routes in the `vpna.inet.0` and `inet.0` routing tables:

```
[edit]
routing-options {
  rib-groups {
    vpna-to-inet0 {
      import-rib [ vpna.inet.0 inet.0 ];
    }
  }
}
```

## Configuring a Routing Instance for Router PE1

On Router PE1, configure a routing instance. As part of the configuration for the routing instance, configure a static route that is installed in `vpna.inet.0` and is pointed at `inet.0` for resolution.

```
[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      static {
        route 0.0.0.0/0 next-table inet.0;
      }
    }
  }
}
```

At the `[edit routing-instances vpna protocols bgp]` hierarchy level, configure a policy (`import-public-addr-to-inet0`) to import public routes into `inet.0` and a routing table group (`vpna-to-inet0`) to allow BGP to install routes into multiple routing tables (`vpna.inet.0` and `inet.0`):

```
[edit routing-instances vpna]
protocols {
  bgp {
    group to-CE1 {
      import import-public-addr-to-inet0;
      family inet {
        unicast {
          rib-group vpna-to-inet0;
        }
      }
    }
  }
}
```

```

        peer-as 63001;
        neighbor 192.168.197.14;
    }
}

```

## Configuring Policy Options for Router PE1

Configure the policy options for Router PE1 to accept all routes initially (**term a**) and then to install routes with a **public-comm** community into routing table inet.0 (**term b**):

```

[edit]
policy-options {
  policy-statement import-public-addr-to-inet0 {
    term a {
      from {
        protocol bgp;
        rib vpna.inet.0;
        community [ public-comm private-comm ];
      }
      then accept;
    }
    term b {
      from {
        protocol bgp;
        community public-comm;
      }
      to rib inet.0;
      then accept;
    }
    term c {
      then reject;
    }
  }
  community private-comm members target:1:333;
  community public-comm members target:1:111;
  community vpna-comm members target:63000:100;
}

```

## Traffic Routed by the Same Interface Bidirectionally (VPN Has Private Addresses): Configuration Summarized by Router

### Router PE1

Routing Options	<pre> [edit] routing-options {   rib-groups {     vpna-to-inet0 {       import-policy import-public-addr-to-inet0;       import-rib [ vpna.inet.0 inet.0 ];     }   } } </pre>
-----------------	--

Routing Instances	<pre> [edit] routing-instances { </pre>
-------------------	---

```

vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
        static {
            route 0.0.0.0/0 next-table inet.0;
        }
    }
}

Routing Instances [edit routing-instances vpna]
Protocols BGP protocols {
    bgp {
        group to-CE1 {
            family inet {
                unicast {
                    rib-group vpna-to-inet0;
                }
            }
        }
        peer-as 63001;
        neighbor 192.168.197.14;
    }
}

Policy Options [edit]
policy-options {
    policy-statement import-public-addr-to-inet0 {
        term a {
            from {
                protocol bgp;
                rib vpna.inet.0;
                community [ public-comm private-comm ];
            }
            then accept;
        }
        term b {
            from {
                protocol bgp;
                community public-comm;
            }
            to rib inet.0;
            then accept;
        }
        term c {
            then reject;
        }
    }
    community private-comm members target:1:333;
    community public-comm members target:1:111;
    community vpna-comm members target:63000:100;
}

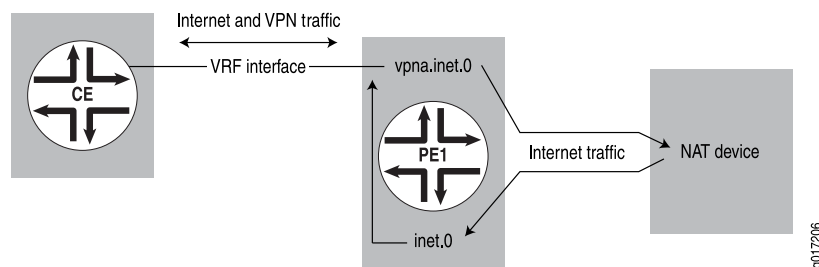
```

}

## Routing Internet Traffic Through a Separate NAT Device

In this example, the CE router does not perform NAT. It sends both VPN and Internet traffic over the same interface to the PE router. The PE router is connected to an NAT device by means of two interfaces. One interface is configured in the PE router's VRF table and points to a VPN interface on the NAT device, which can route Internet traffic for the VPN. The other interface is in a default instance; for example, part of public routing table inet.0. There can be a single physical connection between the PE router and the NAT device and multiple logical connections—one for each VRF table and another interface—as part of the global routing table (see [Figure 50 on page 386](#)).

**Figure 50: Internet Traffic Routed Through a Separate NAT Device**



- [Requirements on page 386](#)
- [Overview on page 386](#)
- [Configuration on page 387](#)

### Requirements

This example uses the following hardware and software components:

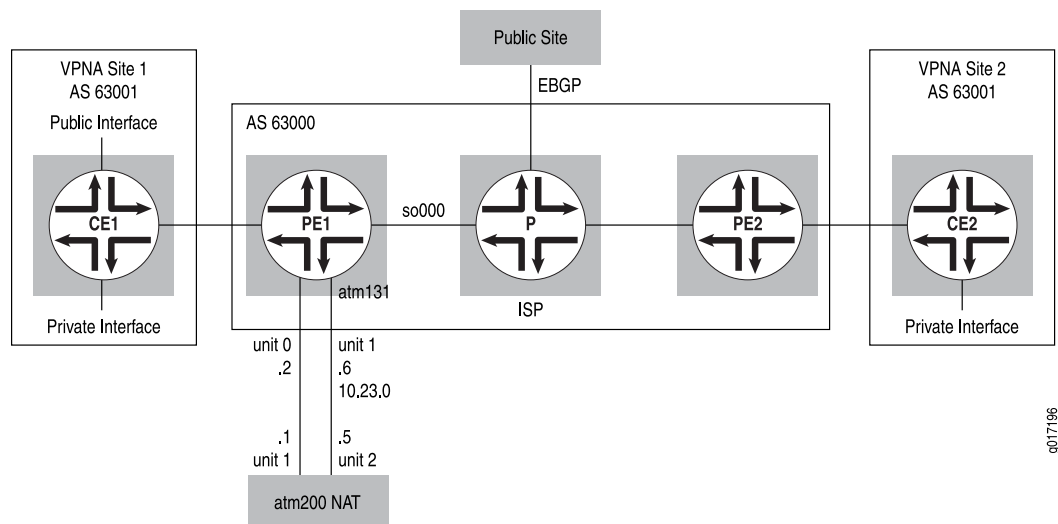
- M Series routers
- Junos OS Release 9.3 or later

### Overview

This example's topology expands upon that illustrated in [Figure 46 on page 371](#). The CE router sends both VPN and Internet traffic to Router PE1. VPN traffic is routed based on the VPN routes received by Router PE1. Traffic for everything else is sent to the NAT device using Router PE1's private interface to the NAT device, which then translates the private addresses and sends the traffic back to Router PE1 using that router's public interface (see [Figure 51 on page 387](#)).

## Topology

Figure 51: Internet Traffic Routed Through a NAT Example Topology



## Configuration

To route Internet traffic through a separate NAT device, perform these tasks:

- [Configuring Interfaces on Router PE1 on page 387](#)
- [Configuring Routing Options for Router PE1 on page 388](#)
- [Configuring Routing Protocols on Router PE1 on page 388](#)
- [Configuring a Routing Instance on Router PE1 on page 389](#)
- [Results on page 391](#)

### Configuring Interfaces on Router PE1

#### Step-by-Step Procedure

1. Configure an interface for VPN traffic from Router CE1:
 

```
[edit]
interfaces {
  t3-0/2/0 {
    dce;
    encapsulation frame-relay;
    unit 0 {
      description "to CE1 VPN interface";
      dlci 10;
      family inet {
        address 192.168.197.13/30;
      }
    }
  }
}
```
2. Configure an interface for VPN traffic to and from the NAT device (unit 0), and an interface for Internet traffic to and from the NAT device (unit 1):

```
[edit]
interfaces {
  at-1/3/1 {
    atm-options {
      vpi 1 maximum-vc 255;
    }
    unit 0 {
      description "to NAT VPN interface";
      vci 1.100;
      family inet {
        address 10.23.0.2/32 {
          destination 10.23.0.1;
        }
      }
    }
    unit 1 {
      description "to NAT public interface";
      vci 1.101;
      family inet {
        address 10.23.0.6/32 {
          destination 10.23.0.5;
        }
      }
    }
  }
}
```

### Configuring Routing Options for Router PE1

- Step-by-Step Procedure**
1. Configure a static route on Router PE1 to direct Internet traffic to the CE router through the NAT device. Router PE1 distributes this route to the Internet.

```
[edit]
routing-options {
  static {
    route 10.12.1.0/24 next-hop 10.23.0.5;
  }
}
```

### Configuring Routing Protocols on Router PE1

- Step-by-Step Procedure**
- Configure the following routing protocols on Router PE1:

1. Configure MPLS on Router PE1. Include the NAT device's VPN interface in the VRF table.

```
[edit]
protocols {
  mpls {
    interface so-0/0/0.0;
    interface at-1/3/1.0;
  }
}
```

2. Configure BGP on Router PE1. Include a policy to advertise the public IP address pool:

```
[edit]
protocols {
  bgp {
    group pe-pe {
      type internal;
      local-address 10.255.14.171;
      family inet {
        any;
      }
      family inet-vpn {
        any;
      }
      export [ fix-nh redist-static ];
      neighbor 10.255.14.177;
      neighbor 10.255.14.173;
    }
  }
}
```

3. Configure IS-IS on Router PE1:

```
[edit]
protocols {
  isis {
    level 1 disable;
    interface so-0/0/0.0;
    interface lo0.0;
  }
}
```

4. Configure LDP on Router PE1:

```
[edit]
protocols {
  ldp {
    interface so-0/0/0.0;
  }
}
```

### Configuring a Routing Instance on Router PE1

#### Step-by-Step Procedure

Configure the Layer 3 VPN routing instance on Router PE1:

1. Configure a routing instance on Router PE1. As part of the routing instance configuration, under **routing-options**, configure a static default route in `vpna.inet.0` pointing to the NAT device's VPN interface (this directs all non-VPN traffic to the NAT device):

```
[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    interface at-1/3/1.0;
```

```

route-distinguisher 10.255.14.171:100;
vrf-import vpna-import;
vrf-export vpna-export;
routing-options {
    static {
        route 0.0.0.0/0 next-hop 10.23.0.1;
    }
}
protocols {
    bgp {
        group to-CE1 {
            peer-as 63001;
            neighbor 192.168.197.14;
        }
    }
}
}
}

```

2. Configure the routing policy for the Layer 3 VPN routing instance on Router PE1:

```

policy-options {
    policy-statement fix-nh {
        then {
            next-hop self;
        }
    }
    policy-statement redist-static {
        term a {
            from {
                protocol static;
                route-filter 10.12.1.0/24 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then accept;
        }
    }
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {

```

```

term a {
    from protocol bgp;
    then {
        community add vpna-comm;
        accept;
    }
}
term b {
    then reject;
}
}
community vpna-comm members target:63000:100;
}

```

## Results

From configuration mode on Router PE1, confirm your configuration by entering the show interfaces, show routing-options, show protocols, show routing-instances and show policy-options commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@PE1# show interfaces
interfaces {
    t3-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 0 {
            description "to CE1 VPN interface";
            dlci 10;
            family inet {
                address 192.168.197.13/30;
            }
        }
    }
    at-1/3/1 {
        atm-options {
            vpi 1 maximum-vcs 255;
        }
        unit 0 {
            description "to NAT VPN interface";
            vci 1.100;
            family inet {
                address 10.23.0.2/32 {
                    destination 10.23.0.1;
                }
            }
        }
        unit 1 {
            description "to NAT public interface";
            vci 1.101;
            family inet {
                address 10.23.0.6/32 {
                    destination 10.23.0.5;
                }
            }
        }
    }
}

```

```
    }  
  }  
}  
  
user@PE1# show routing-options  
routing-options {  
  static {  
    route 10.12.1.0/24 next-hop 10.23.0.5;  
  }  
}  
  
user@PE1# show protocols  
protocols {  
  mpls {  
    interface so-0/0/0.0;  
    interface at-1/3/1.0;  
  }  
  bgp {  
    group pe-pe {  
      type internal;  
      local-address 10.255.14.171;  
      family inet {  
        any;  
      }  
      family inet-vpn {  
        any;  
      }  
      export [ fix-nh redist-static ];  
      neighbor 10.255.14.177;  
      neighbor 10.255.14.173;  
    }  
  }  
  isis {  
    level 1 disable;  
    interface so-0/0/0.0;  
    interface lo0.0;  
  }  
  ldp {  
    interface so-0/0/0.0;  
  }  
}  
  
user@PE1# show routing-instances  
routing-instances {  
  vpna {  
    instance-type vrf;  
    interface t3-0/2/0.0;  
    interface at-1/3/1.0;  
    route-distinguisher 10.255.14.171:100;  
    vrf-import vpna-import;  
    vrf-export vpna-export;  
    routing-options {  
      static {  
        route 0.0.0.0/0 next-hop 10.23.0.1;  
      }  
    }  
    protocols {
```

```
    bgp {
      group to-CE1 {
        peer-as 63001;
        neighbor 192.168.197.14;
      }
    }
  }
}

user@PE1# show policy-options
policy-options {
  policy-statement fix-nh {
    then {
      next-hop self;
    }
  }
  policy-statement redist-static {
    term a {
      from {
        protocol static;
        route-filter 10.12.1.0/24 exact;
      }
      then accept;
    }
    term b {
      from protocol bgp;
      then accept;
    }
    term c {
      then accept;
    }
  }
  policy-statement vpna-import {
    term a {
      from {
        protocol bgp;
        community vpna-comm;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement vpna-export {
    term a {
      from protocol bgp;
      then {
        community add vpna-comm;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
}
```

```

    }
    community vpna-comm members target:63000:100;
  }

```

## Centralized Internet Access

This section describes several ways to configure a CE router to act as a central site for Internet access. Internet traffic from other sites (CE routers) is routed to the hub CE router (which also performs NAT) using that router's VPN interface. The hub CE router then forwards the traffic to a PE router connected to the Internet through another interface identified in the inet.0 table. The hub CE router can advertise a default route to the spoke CE routers. The disadvantage of this type of configuration is that all traffic has to go through the central CE router before going to the Internet, causing network delays if this router receives too much traffic. However, in a corporate network, traffic might have to be routed to a central site because most corporate networks separate the VPN from the Internet by means of a single firewall.

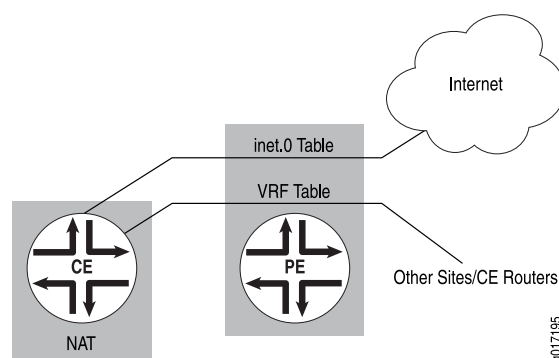
This section includes the following examples:

- [Routing Internet Traffic Through a Hub CE Router on page 394](#)
- [Routing Internet Traffic Through Multiple CE Routers on page 398](#)

### Routing Internet Traffic Through a Hub CE Router

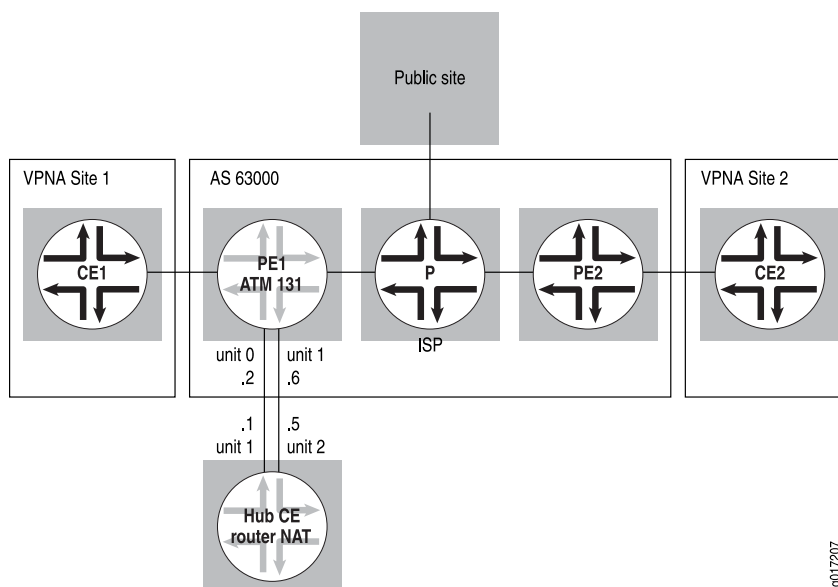
In this example, Internet traffic is routed through a hub CE router. The hub CE router has two interfaces to the hub PE router: a VPN interface and a public interface. It performs NAT on traffic forwarded from the hub PE router through the VPN interface and forwards that traffic from its public interface back to the hub PE router. The hub PE router has a static default route in its VRF table pointing to the hub CE router's VPN interface. It announces this default route to the rest of the VPN, attracting all non-VPN traffic to the hub CE route. The hub PE router also installs and distributes the VPN's public IP address space (see [Figure 52 on page 394](#)).

**Figure 52: Internet Access Through a Hub CE Router Performing NAT**



The configuration for this example is almost identical to that described in [“Routing Internet Traffic Through a Separate NAT Device” on page 386](#). The difference is that Router PE1 is configured to announce a static default route to the other CE routers (see [Figure 53 on page 395](#)).

Figure 53: Internet Access Provided Through a Hub CE Router



The following sections show how to configure centralized Internet access by routing Internet traffic through a hub CE router:

- [Configuring a Routing Instance on Router PE1 on page 395](#)
- [Configuring Policy Options on Router PE1 on page 396](#)
- [Internet Traffic Routed by a Hub CE Router: Configuration Summarized by Router on page 397](#)

### Configuring a Routing Instance on Router PE1

Configure a routing instance for Router PE1. As part of this configuration, under **routing-options**, configure a default static route (**route 0.0.0.0/0**) to be installed in **vpna.inet.0**, and point the route to the hub CE router's VPN interface (**10.23.0.1**). Also, configure BGP under the routing instance to export the default route to the local CE router:

```
[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    interface at-1/3/1.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 10.23.0.1;
      }
    }
  }
}
protocols {
  bgp {
```

```

        group to-CE1 {
            export export-default;
            peer-as 63001;
            neighbor 192.168.197.14;
        }
    }
}
}
}

```

### Configuring Policy Options on Router PE1

Configure policy options on Router PE1. As part of this configuration, Router PE1 should export the static default route to all the remote PE routers in **vpna** (configured in the **policy-statement vpna-export** statement under **term b**):

```

[edit]
policy-options {
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then {
                community add vpna-comm;
                accept;
            }
        }
        term c {
            then reject;
        }
    }
    policy-statement export-default {
        term a {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then reject;
        }
    }
}

```

```
}
```

### Internet Traffic Routed by a Hub CE Router: Configuration Summarized by Router

#### Router PE1

The configuration for Router PE1 is almost identical to that for the example in “[Routing Internet Traffic Through a Separate NAT Device](#)” on page 386. The difference is that Router PE1 is configured to announce a static default route to the other CE routers.

<b>Routing Instance</b>	<pre> routing-instances {   vpna {     instance-type vrf;     interface t3-0/2/0.0;     interface at-1/3/1.0;     route-distinguisher 10.255.14.171:100;     vrf-import vpna-import;     vrf-export vpna-export;     routing-options {       static {         route 0.0.0.0/0 next-hop 10.23.0.1;       }     }     protocols {       bgp {         group to-CE1 {           export export-default;           peer-as 63001;           neighbor 192.168.197.14;         }       }     }   } } </pre>
-------------------------	--

<b>Policy Options</b>	<pre> policy-options {   policy-statement vpna-export {     term a {       from protocol bgp;       then {         community add vpna-comm;         accept;       }     }     term b {       from {         protocol static;         route-filter 0.0.0.0/0 exact;       }       then {         community add vpna-comm;         accept;       }     }     term c {       then reject;     }   } } </pre>
-----------------------	---

```

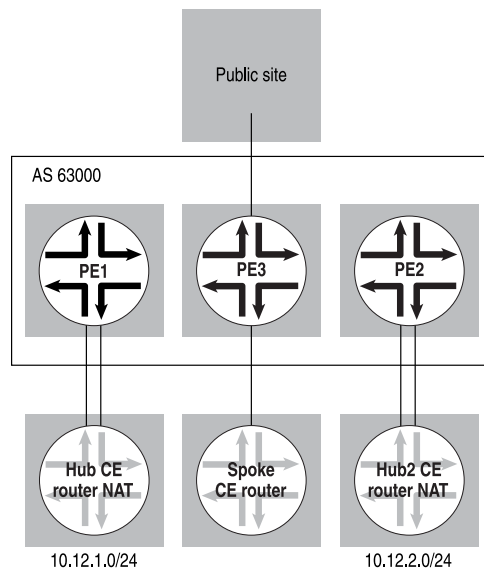
    }
  }
  policy-statement export-default {
    term a {
      from {
        protocol static;
        route-filter 0.0.0.0/0 exact;
      }
      then accept;
    }
    term b {
      from protocol bgp;
      then accept;
    }
    term c {
      then reject;
    }
  }
}

```

## Routing Internet Traffic Through Multiple CE Routers

The example in this section is an extension of that described in “[Centralized Internet Access](#)” on page 394. This example provides different exit points for different sites by means of multiple hub CE routers that perform similar functions. Each hub CE router tags the default route with a different route target and allows the spoke CE routers to select the hub site that should be used for Internet access (see [Figure 54 on page 398](#)).

Figure 54: Two Hub CE Routers Handling Internet Traffic and NAT



This example uses two hub CE routers that handle NAT and Internet traffic:

- Hub1 CE router tags **0/0** with community **public-comm1** (target: 1:111)
- Hub2 CE router tags **0/0** with community **public-comm2** (target: 1:112)

The spoke CE router in this example is configured to have a bias toward Hub2 for Internet access.

The following sections describe how to configure two hub CE routers to handle Internet traffic and NAT:

- [Configuring a Routing Instance on Router PE1 on page 399](#)
- [Configuring Policy Options on Router PE1 on page 399](#)
- [Configuring a Routing Instance on Router PE3 on page 400](#)
- [Configuring Policy Options on Router PE3 on page 401](#)
- [Routing Internet Traffic Through Multiple CE Routers: Configuration Summarized by Router on page 402](#)

### Configuring a Routing Instance on Router PE1

Configure a routing instance on Router PE1:

```
[edit]
routing-instances {
  vpna {
    instance-type vrf;
    interface t3-0/2/0.0;
    interface at-1/3/1.0;
    route-distinguisher 10.255.14.171:100;
    vrf-import vpna-import;
    vrf-export vpna-export;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 10.23.0.1;
      }
    }
    protocols {
      bgp {
        group to-CE1 {
          export export-default;
          peer-as 63001;
          neighbor 192.168.197.14;
        }
      }
    }
  }
}
```

### Configuring Policy Options on Router PE1

The policy options for Router PE1 are the same as in “[Routing Internet Traffic Through a Hub CE Router](#)” on page 394, but the configuration in this example includes an additional community, **public-comm1**, in the **export** statement:

```
[edit]
policy-options {
  policy-statement vpna-import {
    term a {
      from {
```

```

        protocol bgp;
        community vpna-comm;
    }
    then accept;
}
term b {
    then reject;
}
}
policy-statement vpna-export {
    term a {
        from {
            protocol static;
            route-filter 0.0.0.0/0 exact;
        }
        then {
            community add public-comm1;
            community add vpna-comm;
            accept;
        }
    }
    term b {
        from protocol bgp;
        then {
            community add vpna-comm;
            accept;
        }
    }
    term c {
        then reject;
    }
}
community public-comm1 members target:1:111;
community public-comm2 members target:1:112;
community vpna-comm members target:63000:100;
}

```

The configuration of Router PE2 is identical to that of Router PE1 except that Router PE2 exports the default route through community **public-comm2**.

### Configuring a Routing Instance on Router PE3

Configure routing instance **vpna** on Router PE3:

```

[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t1-0/2/0.0;
        route-distinguisher 10.255.14.173:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            rip {
                group to-vpn12 {
                    export export-CE;
                }
            }
        }
    }
}

```

```

        neighbor t1-0/2/0.0;
    }
}
}
}

```

### Configuring Policy Options on Router PE3

Configure the **vrf-import** policy for Router PE3 to select the Internet exit point based on the additional communities specified in [“Configuring Policy Options on Router PE1” on page 399](#):

```

[edit]
policy-options {
  policy-statement vpna-export {
    term a {
      from protocol rip;
      then {
        community add vpna-comm;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  policy-statement vpna-import {
    term a {
      from {
        protocol bgp;
        community public-comm1;
        route-filter 0.0.0.0/0 exact;
      }
      then reject;
    }
    term b {
      from {
        protocol bgp;
        community vpna-comm;
      }
      then accept;
    }
    term c {
      then reject;
    }
  }
  policy-statement export-CE {
    from protocol bgp;
    then accept;
  }
  community vpna-comm members target:69:100;
  community public-comm1 members target:1:111;
  community public-comm2 members target:1:112;
}

```

## Routing Internet Traffic Through Multiple CE Routers: Configuration Summarized by Router

### Router PE1

This configuration is an extension of the example in “[Routing Internet Traffic Through a Hub CE Router](#)” on page 394. It provides different exit points for various sites by using multiple hub CE routers that perform similar functions.

<b>Routing Instances</b>	<pre> routing-instances {   vpn {     instance-type vrf;     interface t3-0/2/0.0;     interface at-1/3/1.0;     route-distinguisher 10.255.14.171:100;     vrf-import vpn-import;     vrf-export vpn-export;     routing-options {       static {         route 0.0.0.0/0 next-hop 10.23.0.1;       }     }     protocols {       bgp {         group to-CE1 {           export export-default;           peer-as 63001;           neighbor 192.168.197.14;         }       }     }   } } </pre>
--------------------------	---

<b>Policy Options</b>	<pre> policy-options {   policy-statement vpn-import {     term a {       from {         protocol bgp;         community vpn-comm;       }       then accept;     }     term b {       then reject;     }   }   policy-statement vpn-export {     term a {       from {         protocol static;         route-filter 0.0.0.0/0 exact;       }       then {         community add public-comm1;       }     }   } } </pre>
-----------------------	--

```

        community add vpna-comm;
        accept;
    }
}
term b {
    from protocol bgp;
    then {
        community add vpna-comm;
        accept;
    }
}
term c {
    then reject;
}
}
community public-comm1 members target:1:111;
community public-comm2 members target:1:112;
community vpna-comm members target:63000:100;
}

```

### Router PE2

The configuration of Router PE2 is identical to that of Router PE1, except that Router PE2 exports the default route through community **public-comm2** (see [“Policy Options” on page 402](#)).

### Router PE3

<b>Routing Instances</b>	<pre> routing-instances {     vpna {         instance-type vrf;         interface t1-0/2/0.0;         route-distinguisher 10.255.14.173:100;         vrf-import vpna-import;         vrf-export vpna-export;         protocols {             rip {                 group to-vpn12 {                     export export-CE;                     neighbor t1-0/2/0.0;                 }             }         }     } } </pre>
--------------------------	---

<b>Policy Options</b>	<pre> policy-options {     policy-statement vpna-export {         term a {             from protocol rip;             then {                 community add vpna-comm;                 accept;             }         }         term b { </pre>
-----------------------	---

```
        then reject;
    }
}
policy-statement vpn-import {
    term a {
        from {
            protocol bgp;
            community public-comm1;
            route-filter 0.0.0.0/0 exact;
        }
        then reject;
    }
    term b {
        from {
            protocol bgp;
            community vpn-comm;
        }
        then accept;
    }
    term c {
        then reject;
    }
}
policy-statement export-CE {
    from protocol bgp;
    then accept;
}
community vpn-comm members target:69:100;
community public-comm1 members target:1:111;
community public-comm2 members target:1:112;
}
```

## CHAPTER 6

# Layer 3 VPN Additional Examples

- [Example: Configuring Interprovider Layer 3 VPN Option A on page 405](#)
- [Example: Configuring Interprovider Layer 3 VPN Option B on page 425](#)
- [Example: Configuring Interprovider Layer 3 VPN Option C on page 444](#)
- [Layer 3 VPN Overview on page 466](#)
- [Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 468](#)
- [Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview on page 468](#)
- [Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN on page 470](#)
- [Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 493](#)

### Example: Configuring Interprovider Layer 3 VPN Option A

---

Interprovider Layer 3 VPN Option A provides interprovider VRF-to-VRF connections at the AS boundary routers (ASBRs). Compared to Option B and Option C, Option A is the least scalable solution.

This example provides a step-by-step procedure to configure interprovider Layer 3 VPN option A, which is one of the recommended implementations of MPLS VPN when that service is required by a customer that has more than one AS and but not all of the customer's ASs can be serviced by the same service provider. It is organized in the following sections:

- [Requirements on page 405](#)
- [Overview and Topology on page 406](#)
- [Configuration on page 407](#)

### Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.5 or later.
- Eight M Series, T Series, TX Series, or MX Series Juniper Networks routers.

## Overview and Topology

This is the simplest and least scalable interprovider VPN solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same service provider (SP).

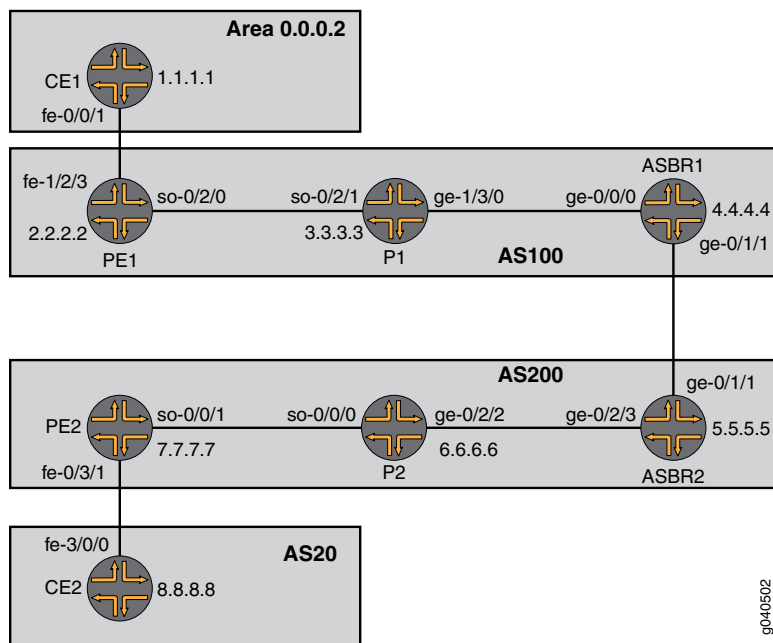
*RFC 4364*, section 10, refers to this method as Interprovider VRF-to-VRF connections at the AS border routers.

In this configuration:

- The virtual routing and forwarding (VRF) table in the ASBR of one AS is linked to the VRF table in the ASBR in the other AS. Each ASBR must contain a VRF instance for every VPN configured in both service provider networks. Then an IGP or BGP must be configured between the ASBRs. This has the disadvantage of limiting scalability.
- In this configuration, the autonomous system boundary routers (ASBRs) at both SPs are configured as regular PE routers, and provide MPLS L3 VPN service to the neighbor SP.
- Each PE router treats the other as if it were a customer edge (CE) router. ASBRs play the role of regular CE routers for the ASBR of the remote SP. ASBRs see each other as CE devices.
- A provider edge (PE) router in one autonomous system (AS) attaches directly to a PE router in another AS.
- The two PE routers are attached by multiple sub-interfaces, at least one for each of the VPNs whose routes need to be passed from AS to AS.
- The PE routers associate each sub-interface with a VPN routing and forwarding (VRF) table, and use EBGp to distribute unlabeled IPv4 addresses to each other.
- In this solution, all common VPNs defined at both PEs must also be defined at one or more ASBRs between the two SPs. This is not a very scalable methodology, especially when a transit SP is used by two regional SPs for interconnection.
- This is a procedure that is simple to configure and it does not require MPLS at the border between ASs. Additionally, it does not scale as well as other recommended procedures.

The topology of the network is shown in [Figure 55 on page 407](#).

Figure 55: Physical Topology of Interprovider Layer 3 VPN Option A



## Configuration



**NOTE:** The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure interprovider layer 3 VPN option A, perform the following tasks:

- [Configuring Router CE1 on page 407](#)
- [Configuring Router PE1 on page 408](#)
- [Configuring Router P1 on page 411](#)
- [Configuring Router ASBR1 on page 412](#)
- [Configuring Router ASBR2 on page 414](#)
- [Configuring Router P2 on page 416](#)
- [Configuring Router PE2 on page 417](#)
- [Configuring Router CE2 on page 419](#)
- [Verifying the VPN Operation on page 420](#)

### Configuring Router CE1

#### Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE1 and Router PE1. Specify the **inet** address family type.

```
[edit interfaces fe-0/0/1.0]
family inet {
  address 18.18.18.1/30;
}
```

2. On Router CE1, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```
[edit interfaces lo0]
unit 0 {
  family inet {
    address 1.1.1.1/32;
  }
}
```

3. On Router CE1, configure an IGP. The IGP can be a static route, RIP, OSPF, ISIS, or EBGp. In this example we configure OSPF. Include the Fast Ethernet interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1.

```
[edit protocols]
ospf {
  area 0.0.0.2 {
    interface fe-0/0/1.0;
    interface lo0.0;
  }
}
```

### Configuring Router PE1

#### Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
so-0/2/0 {
  unit 0 {
    family inet {
      address 19.19.19.1/30;
    }
    family mpls;
  }
}
fe-1/2/3 {
  unit 0 {
    family inet {
      address 18.18.18.2/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 2.2.2.2/32;
    }
  }
}
```

```
    }
  }
```

2. On Router PE1, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing Fast Ethernet interface and specify the export policy to export BGP routes into OSPF.

```
[edit routing-instances]
vpn2CE1 {
  instance-type vrf;
  interface fe-1/2/3.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    ospf {
      export bgp-to-ospf;
      area 0.0.0.2 {
        interface fe-1/2/3.0;
      }
    }
  }
}
```

3. On Router PE1, configure the RSVP and MPLS protocols to support the label-switched path (LSP). Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure a BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the **inet-vpn** address family and **unicast** traffic type to enable BGP to carry IPv4 network layer reachability information (NLRI) for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE1.

```
[edit protocols]
rsvp {
  interface so-0/2/0.0;
  interface lo0.0;
}
mpls {
  label-switched-path To-ASBR1 {
    to 4.4.4.4;
  }
  interface so-0/2/0.0;
  interface lo0.0;
}
bgp {
  group To_ASBR1 {
    type internal;
    local-address 2.2.2.2;
    neighbor 4.4.4.4 {
      family inet-vpn {
```

```
        unicast;
    }
}
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-0/2/0.0;
        interface lo0.0;
    }
}
```

4. On Router PE1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

5. On Router PE1, configure a policy to export the BGP routes into OSPF.

```
[edit policy-options]
policy-statement bgp-to-ospf {
    term 1 {
        from protocol bgp;
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

6. On Router PE1, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
    term 1 {
        from protocol ospf;
        then {
            community add test_comm;
            accept;
        }
    }
    term 2 {
        then reject;
    }
}
```

7. On Router PE1, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```
[edit policy-options]
policy-statement vpnimport {
    term 1 {
        from {
            protocol bgp;
            community test_comm;
        }
    }
}
```

```

        then accept;
    }
    term 2 {
        then reject;
    }
}

```

8. On Router PE1, define the **test\_comm** BGP community with a route target.

```

[edit policy-options]
community test_comm members target:1:100;

```

### Configuring Router P1

#### Step-by-Step Procedure

1. On Router P1, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
so-0/2/1 {
    unit 0 {
        family inet {
            address 19.19.19.2/30;
        }
        family mpls;
    }
}
ge-1/3/0 {
    unit 0 {
        family inet {
            address 20.20.20.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 3.3.3.3/32;
        }
    }
}

```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface so-0/2/1.0;
    interface ge-1/3/0.0;
    interface lo0.0;
}

```

```

}
mpls {
  interface lo0.0;
  interface ge-1/3/0.0;
  interface so-0/2/1.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-1/3/0.0;
    interface so-0/2/1.0;
    interface lo0.0;
  }
}

```

### Configuring Router ASBR1

#### Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** addresses families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
ge-0/0/0 {
  unit 0 {
    family inet {
      address 20.20.20.2/30;
    }
    family mpls;
  }
}
ge-0/1/1 {
  unit 0 {
    family inet {
      address 21.21.21.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 4.4.4.4/32;
    }
  }
}

```

2. On Router ASBR1, configure the **To\_ASBR2** routing instance. Specify the **vrf** instance type and specify the core-facing Gigabit Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Configure a route target for the VPN. Configure the BGP peer group within the VRF. Specify AS 200 as the peer AS and specify the IP address of the Gigabit Ethernet interface on Router ASBR2 as the neighbor address.

```

[edit routing instances]

```

```

To_ASBR2{
  instance-type vrf;
  interface ge-0/1/1.0;
  route-distinguisher 1:100;
  vrf-target target:1:100;
  protocols {
    bgp {
      group To_ASBR2 {
        type external;
        neighbor 21.21.21.2 {
          peer-as 200;
        }
      }
    }
  }
}

```

3. On Router ASBR1, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
  interface ge-0/0/0.0;
  interface lo0.0;
}
mpls {
  label-switched-path To_PE1 {
    to 2.2.2.2;
  }
  interface lo0.0;
  interface ge-0/0/0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-0/0/0.0;
    interface lo0.0;
  }
}

```

4. On Router ASBR1, create the **To-PE1** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the **lo0.0** interface address of Router PE1.

```

[edit protocols]
bgp {
  group To-PE1 {
    type internal;
    local-address 4.4.4.4;
    neighbor 2.2.2.2 {
      family inet-vpn {
        unicast;
      }
    }
  }
}

```

```

    }
  }
}

```

5. On Router ASBR1, configure the BGP local autonomous system number.

```

[edit routing-options]
autonomous-system 100;

```

### Configuring Router ASBR2

#### Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
ge-0/1/1 {
  unit 0 {
    family inet {
      address 21.21.21.2/30;
    }
    family mpls;
  }
}
ge-0/2/3 {
  unit 0 {
    family inet {
      address 22.22.22.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 5.5.5.5/32;
    }
  }
}

```

2. On Router ASBR2, configure the **To\_ASBR1** routing instance. Specify the **vrf** instance type and specify the core-facing Gigabit Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Configure a route target for the VPN. Configure the BGP peer group within the VRF. Specify AS 100 as the peer AS and specify the IP address of the Gigabit Ethernet interface on Router ASBR1 as the neighbor address.

```

[edit routing-instances]
To_ASBR1 {
  instance-type vrf;
  interface ge-0/1/1.0;
  route-distinguisher 1:100;
  vrf-target target:1:100;
  protocols {
    bgp {

```

```

        group To_ASBR1 {
            type external;
            neighbor 21.21.21.1 {
                peer-as 100;
            }
        }
    }
}

```

3. On Router ASBR2, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface ge-0/2/3.0;
    interface lo0.0;
}
mpls {
    label-switched-path To_PE2 {
        to 7.7.7.7;
    }
    interface lo0.0;
    interface ge-0/2/3.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/3.0;
        interface lo0.0;
    }
}

```

4. On Router ASBR2, create the **To-PE2** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the **lo0.0** interface address of Router PE2.

```

[edit protocols]
bgp {
    group To-PE2 {
        type internal;
        local-address 5.5.5.5;
        neighbor 7.7.7.7 {
            family inet-vpn {
                unicast;
            }
        }
    }
}

```

5. On Router ASBR2, configure the BGP local autonomous system number.

```

[edit routing-options]
autonomous-system 200;

```

## Configuring Router P2

### Step-by-Step Procedure

1. On Router P2, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```
[edit interfaces]
so-0/0/0 {
  unit 0 {
    family inet {
      address 23.23.23.1/30;
    }
    family mpls;
  }
}
ge-0/2/2 {
  unit 0 {
    family inet {
      address 22.22.22.2/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 6.6.6.6/32;
    }
  }
}
```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
  interface so-0/0/0.0;
  interface ge-0/2/2.0;
  interface lo0.0;
}
mpls {
  interface lo0.0;
  interface ge-0/2/2.0;
  interface so-0/0/0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-0/2/2.0;
    interface so-0/0/0.0;
    interface lo0.0;
  }
}
```

```
}
}
```

### Configuring Router PE2

#### Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
so-0/0/1 {
  unit 0 {
    family inet {
      address 23.23.23.2/30;
    }
    family mpls;
  }
}
fe-0/3/1 {
  unit 0 {
    family inet {
      address 24.24.24.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 7.7.7.7/32;
    }
  }
}
```

2. On Router PE2, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the BGP peer group within the VRF. Specify AS **20** as the peer AS and specify the IP address of the Fast Ethernet interface on Router CE2 as the neighbor address.

```
[edit routing-instances]
vpn2CE2 {
  instance-type vrf;
  interface fe-0/3/1.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    bgp {
      group To_CE2 {
        peer-as 20;
        neighbor 24.24.24.2;
      }
    }
  }
}
```

```
}
```

3. On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure a BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2. Specify the **inet-vpn** address family and **unicast** traffic type to enable BGP to carry IPv4 NLRI for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE2.

```
[edit protocols]
rsvp {
  interface so-0/0/1.0;
  interface lo0.0;
}
mpls {
  label-switched-path To-ASBR2 {
    to 5.5.5.5;
  }
  interface so-0/0/1.0;
  interface lo0.0;
}
bgp {
  group To_ASBR2 {
    type internal;
    local-address 7.7.7.7;
    neighbor 5.5.5.5 {
      family inet-vpn {
        unicast;
      }
    }
  }
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface so-0/0/1.0;
    interface lo0.0;
  }
}
```

4. On Router PE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

5. On Router PE2, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
  term 1 {
    from protocol bgp;
    then {
      community add test_comm;
      accept;
    }
  }
}
```

```

    }
  }
  term 2 {
    then reject;
  }
}

```

6. On Router PE2, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```

[edit policy-options]
policy-statement vpnimport {
  term 1 {
    from {
      protocol bgp;
      community test_comm;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}

```

7. On Router PE2, define the **test\_comm** BGP community with a route target.

```

[edit policy-options]
community test_comm members target:1:100;

```

### Configuring Router CE2

#### Step-by-Step Procedure

1. On Router CE2, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE2 and Router PE2. Specify the **inet** address family type.

```

[edit interfaces]
fe-3/0/0 {
  unit 0 {
    family inet {
      address 24.24.24.2/30;
    }
  }
}

```

2. On Router CE2, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```

[edit interfaces lo0]
lo0 {
  unit 0 {
    family inet {
      address 8.8.8.8/32;
    }
  }
}

```

- On Router CE2, configure an IGP. The IGP can be a static route, RIP, OSPF, ISIS, or EBGP. In this example, we configure EBGP. Specify AS **200** as the peer AS and specify the BGP neighbor IP address as the Fast Ethernet interface of Router PE2.

```
[edit protocols]
bgp {
  group To_PE2 {
    neighbor 24.24.24.1 {
      export myroutes;
      peer-as 200;
    }
  }
}
```

### Verifying the VPN Operation

#### Step-by-Step Procedure

- Commit the configuration on each router.



**NOTE:** The MPLS labels shown in this example will be different than the labels used in your configuration.

- On Router PE1, display the routes for the **vpn2CE1** routing instance using the **show ospf route** command. Verify that the 1.1.1.1 route is learned from OSPF.

```
user@PE1> show ospf route instance vpn2CE1
```

Topology default Route Table:

Prefix	Path	Route	NH	Metric	NextHop	Nexthop
	Type	Type	Type		Interface	addr/label
1.1.1.1	Intra	Router	IP	1	fe-1/2/3.0	18.18.18.1
1.1.1.1/32	Intra	Network	IP	1	fe-1/2/3.0	18.18.18.1
18.18.18.0/30	Intra	Network	IP	1	fe-1/2/3.0	18.18.18.1

- On Router PE1, use the **show route advertising-protocol** command to verify that Router PE1 advertises the 1.1.1.1 route to Router ASBR1 using MP-BGP with the VPN MPLS label.

```
user@PE1> show route advertising-protocol bgp 4.4.4.4 extensive
```

vpn2CE1.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

\* 1.1.1.1/32 (1 entry, 1 announced)

```
BGP group To_PE1 type Internal
Route Distinguisher: 1:100
VPN Label: 299856
Nexthop: Self
Flags: Nexthop Change
MED: 1
Localpref: 100
AS path: [100] I
Communities: target:1:100 rte-type:0.0.0.2:1:0
```

- On Router ASBR1, use the **show route receive-protocol** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **To\_ASBR2.inet.0** routing table.

```

user@ASBR1> show route receive-protocol bgp 2.2.2.2 extensive
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1/32 (1 entry, 1 announced)
  Route Distinguisher: 1:100
  VPN Label: 299856
  Nexthop: 2.2.2.2
  MED: 1
  Localpref: 100
  AS path: I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

MPLS.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

BGP.13VPN.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 1:100:1.1.1/32 (1 entry, 0 announced)
  Route Distinguisher: 1:100
  VPN Label: 299856
  Nexthop: 2.2.2.2
  MED: 1
  Localpref: 100
  AS path: I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

5. On Router ASBR1, use the **show route advertising-protocol bgp 21.21.21.2 extensive** command to verify that Router ASBR1 advertises the 1.1.1.1 route to Router ASBR2.

```

user@ASBR1> show route advertising-protocol bgp 21.21.21.2 extensive
To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1/32 (1 entry, 1 announced)
  BGP group To_ASBR2.inet.0 type External
  Nexthop: Self
  AS path: [100] I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

6. On Router ASBR2, use the **show route receive-protocol bgp 21.21.21.1 extensive** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **To\_ASBR1.inet.0** routing table.

```

user@ASBR2> show route receive-protocol bgp 21.21.21.1 extensive
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1/32 (1 entry, 1 announced)
  Accepted
  Nexthop: 21.21.21.1
  AS path: 100 I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

```

MPLS.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

BGP.13VPN.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

```

7. On Router ASBR2, use the **show route advertising-protocol bgp 1.1.1.1 extensive** command to verify that Router ASBR2 advertises the 1.1.1.1 route to Router PE2 in the **To-PE2** routing instance.

```

user@ASBR2> show route advertising-protocol bgp 7.7.7.7 extensive
To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  BGP group To-PE2 type Internal
    Route Distinguisher: 1:100
    VPN Label: 299936
    Nexthop: Self
    Flags: Nexthop Change
    Localpref: 100
    AS path: [200] 100 I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

8. On Router PE2, use the **show route receive-protocol** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **To\_CE2.inet.0** routing table.

```

user@PE2> show route receive-protocol bgp 5.5.5.5 extensive
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

__juniper_private1__.inet.0: 14 destinations, 14 routes (8 active, 0 holddown,
6 hidden)

__juniper_private2__.inet.0: 1 destinations, 1 routes (0 active, 0 holddown,
1 hidden)

To_CE2.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  Accepted
  Route Distinguisher: 1:100
  VPN Label: 299936
  Nexthop: 5.5.5.5
  Localpref: 100
  AS path: 100 I
  AS path: Recorded
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

9. On Router PE2, use the **show route advertising-protocol** command to verify that Router PE2 advertises the 1.1.1.1 route to Router CE2 through the **To\_CE2** peer group.

```

user@PE2> show route advertising-protocol bgp 24.24.24.2 extensive
To_CE2.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  BGP group To_CE2 type External
    Nexthop: Self
    AS path: [200] 100 I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

10. On Router CE2, use the **show route** command to verify that Router CE2 receives the 1.1.1.1 route from Router PE2.

```

user@CE2> show route 1.1.1.1
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32          *[BGP/170] 00:25:36, localpref 100
                    AS path: 200 100 I
                    > to 24.24.24.1 via fe-3/0/0.0

```

11. On Router CE2, use the **ping** command and specify **8.8.8.8** as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 1.1.1.1 source 8.8.8.8
PING 1.1.1.1 (1.1.1.1): 56 data bytes
64 bytes from 1.1.1.1: icmp_seq=0 ttl=58 time=4.672 ms
64 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=10.480 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=10.560 ms
```

12. On Router PE2, use the **show route** command to verify that the traffic is sent with an inner label of **299936** and a top label of **299776**.

```
user@PE2> show route 1.1.1.1 detail

To_CE2.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
1.1.1./32 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
              Route Distinguisher: 1:100
              Next hop type: Indirect
              Next-hop reference count: 6
              Source: 5.5.5.5
              Next hop type: Router, Next hop index: 648
              Next hop: via so-0/0/1.0 weight 0x1, selected
              Label-switched-path To-ASBR2
              Label operation: Push 299936, Push 299776(top)
              Protocol next hop: 5.5.5.5
              Push 299984
              Indirect next hop: 8c6109c 262143
              State: <Secondary Active Int Ext>
              Local AS: 200 Peer AS: 200
              Age: 3:37 Metric2: 2
              Task: BGP_200.5.5.5+179
              Announcement bits (3): 0-RT 1-KRT 2-BGP RT Background
              AS path: 100 I
              AS path: Recorded
              Communities: target:1:100 rte-type:0.0.0.2:1:0
              Accepted
              VPN Label: 299984
              Localpref: 100
              Router ID: 5.5.5.5
              Primary Routing Table BGP.13VPN.0
```

13. On Router ASBR2, use the **show route table** command to verify that Router ASBR2 receives the traffic.

```
1ab@ASBR2# show route table mpls.0 detail
299936 (1 entry, 1 announced)
  *VPN      Preference: 170
              Next hop type: Router, Next hop index: 649
              Next-hop reference count: 2
              Source: 21.21.21.1
              Next hop: 21.21.21.1 via ge-0/1/1.0, selected
              Label operation: Pop
              State: <Active Int Ext>
              Local AS: 200
              Age: 9:54
              Task: BGP RT Background
              Announcement bits (1): 0-KRT
              AS path: 100 I
              Ref Cnt: 1
```

Communities: target:1:100 rte-type:0.0.0.2:1:0

14. On Router ASBR2, use the **show route table** command to verify that Router ASBR2 receives the traffic.

```
lab@ASBR2# show route 1.1.1.1 detail
To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
1.1.1.1/32 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Next hop type: Router, Next hop index: 576
            Next-hop reference count: 3
            Source: 21.21.21.1
            Next hop: 21.21.21.1 via ge-0/1/1.0, selected
            State: <Active Ext>
            Peer AS: 100
            Age: 13:07
            Task: BGP_100.21.21.21.1+53372
            Announcement bits (2): 0-KRT 1-BGP RT Background
            AS path: 100 I
            Communities: target:1:100 rte-type:0.0.0.2:1:0
            Accepted
            Localpref: 100
            Router ID: 21.21.21.1
```

15. On Router ASBR1, use the **show route** command to verify that ASBR1 sends traffic toward PE1 with the top label **299792** and VPN label **299856**.

```
lab@ASBR1# show route 1.1.1.1 detail
To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
1.1.1.1/32 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Route Distinguisher: 1:100
            Next hop type: Indirect
            Next-hop reference count: 3
            Source: 2.2.2.2
            Next hop type: Router, Next hop index: 669
            Next hop: 20.20.20.1 via ge-0/0/0.0 weight 0x1, selected
            Label-switched-path To_PE1
            Label operation: Push 299856, Push 299792(top)
            Protocol next hop: 2.2.2.2                      Push 299856
            Indirect next hop: 8af70a0 262143
            State: <Secondary Active Int Ext>
            Local AS: 100 Peer AS: 100
            Age: 12:15      Metric: 1      Metric2: 2
            Task: BGP_100.2.2.2.2+58065
            Announcement bits (2): 0-KRT 1-BGP RT Background
            AS path: I
            Communities: target:1:100 rte-type:0.0.0.2:1:0
            VPN Label: 299856
            Localpref: 100
            Router ID: 2.2.2.2
            Primary Routing Table BGP.13VPN.0
```

16. On Router PE1, use the **show route table** command to verify that Router PE1 receives the traffic with label **299856**, pops the label, and the traffic is sent toward Router CE1 through interface **fe-1/2/3.0**.

```
lab@PE1# show route table mpls.0 detail
299856 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Router, Next hop index: 666
```

```

Next-hop reference count: 2
Next hop: 18.18.18.1 via fe-1/2/3.0, selected
Label operation: Pop
State: <Active Int Ext>
Local AS: 100
Age: 17:38
Task: BGP RT Background
Announcement bits (1): 0-KRT
AS path: I
Ref Cnt: 1
Communities: rte-type:0.0.0.2:1:0

```

17. On Router PE1, use the **show route** command to verify that PE1 receives the traffic after the top label is popped by Router P and the traffic is sent toward Router CE1 through interface **fe-1/2/3.0**.

```
lab@PE1# show route 1.1.1.1 detail
```

```

vpn2CE1.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
1.1.1.1/32 (1 entry, 1 announced)
  *OSPF Preference: 10
    Next hop type: Router, Next hop index: 634
    Next-hop reference count: 3
    Next hop: 18.18.18.1 via fe-1/2/3.0, selected
    State: <Active Int>
    Age: 18:42 Metric: 1
    Area: 0.0.0.2
    Task: VPN2alice-OSPFv2
    Announcement bits (2): 2-KRT 3-BGP RT Background
    AS path: I
    Communities: rte-type:0.0.0.2:1:0

```

## Related Documentation

### Example: Configuring Interprovider Layer 3 VPN Option B

Interprovider Layer 3 VPN Option B provides interprovider EBGp redistribution of labeled VPN-IPv4 routes from AS to neighboring AS. This solution is considered to be more scalable than Option A, but not as scalable as Option C.

This example provides a step-by-step procedure to configure interprovider layer 3 VPN option B, which is one of the recommended implementations of an MPLS VPN when that service is required by a customer that has more than one AS, but not all of the customer's ASs can be serviced by the same service provider. It is organized in the following sections:

- [Requirements on page 425](#)
- [Configuration Overview and Topology on page 426](#)
- [Configuration on page 427](#)

## Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.5 or later.

- Eight M Series, T Series, TX Series, or MX Series Juniper Networks routers.

## Configuration Overview and Topology

Interprovider layer 3 VPN option B is a somewhat scalable solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same service provider. *RFC 4364*, section 10, refers to this method as interprovider EBGp redistribution of labeled VPN-IPv4 routes from AS to neighboring AS.

In the topology shown in Figure 1, the following events occur:

- The PE routers use IBGP to redistribute labeled VPN-IPv4 routes either to an ASBR, or to a route reflector of which an ASBR is a client.
- The ASBR then uses EBGp to redistribute those labeled VPN-IPv4 routes to an ASBR in another AS, which distributes them to the PE routers in that AS, or to another ASBR for distribution.
- Labeled VPN-IPv4 routes are distributed between ASBR routers on each site. There is no need to define a separate VPN routing and forwarding instance (VRF) for each common VPN that resides on two different SPs.
- Router PE2 distributes VPN-IPv4 routes to Router ASBR2 using MP-IBGP.
- Router ASBR2 distributes these labeled VPN-IPv4 routes to Router ASBR1, using the MP-EBGP session between them.
- Router ASBR1 redistributes those routes to Router PE1, using MP-IBGP. Each time a label is advertised, routers change the next-hop information and labels.
- An MPLS path is established between Router PE1 and Router PE2. This path enables changing of the next-hop attribute for the routes that are learned from the neighbor SP router and map the incoming label for the given routes to the outgoing label advertised to PE routers in the internal network.
- The ingress PE router inserts two labels onto the IP packet coming from the end customer. The inner label is for the VPN-IPv4 routes learned from internal ASBRs and the outer label is for the route to the internal ASBR, obtained through resource reservation protocol (RSVP) or label distribution protocol (LDP).
- When a packet arrives at the ASBR, it removes the outer label (when explicit-null signaling is used; otherwise, penultimate hop-popping (PHP) pops the label) and swaps the inner label with the label obtained from the neighbor ASBR through MP-EBGP label and prefix advertisements.
- The second ASBR swaps the VPN-IPv4 label and pushes another label to reach the PE router in its own AS.
- The remaining process is the same as for a regular VPN.

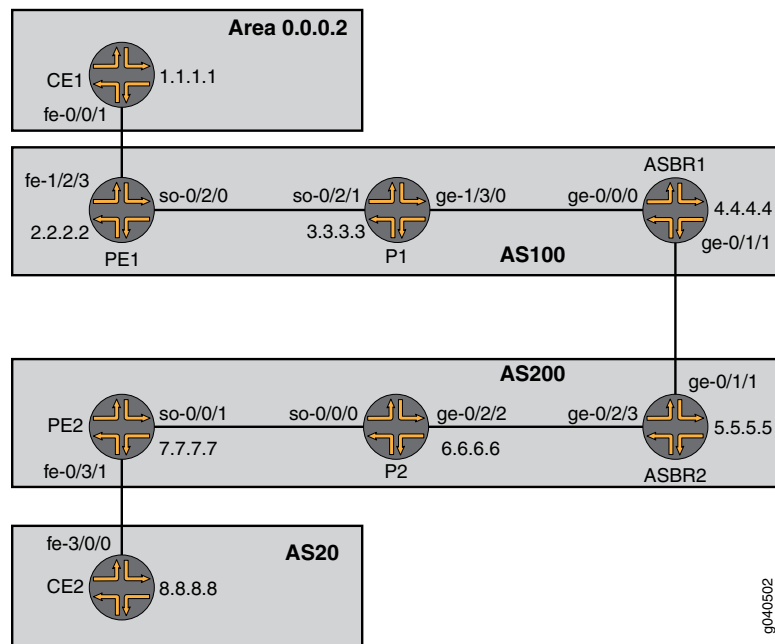


**NOTE:** In this solution, ASBR routers keep all VPN-IPv4 routes in the routing information base (RIB), and the labels associated with the prefixes are kept in the forwarding information base (FIB). Because the RIB and FIB tables can take occupy much of the respective allocated memory, this solution is not very scalable for an interprovider VPN.

If a transit SP is used between SP1 and SP2, the transit SP also has to keep all VPN-IPv4 routes in the RIB and the corresponding labels in the FIB. The ASBRs at the transit SP have the same functionality as ASBRs in the SP1 or SP2 networks in this solution.

The topology of the network is shown in [Figure 56 on page 427](#).

**Figure 56: Physical Topology of Interprovider Layer 3 VPN Option B**



## Configuration



**NOTE:** The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure layer 3 VPN option B, perform the following tasks:

- [Configuring Router CE1 on page 428](#)
- [Configuring Router PE1 on page 428](#)
- [Configuring Router P1 on page 431](#)

- [Configuring Router ASBR1 on page 432](#)
- [Configuring Router ASBR2 on page 434](#)
- [Configuring Router P2 on page 435](#)
- [Configuring Router PE2 on page 436](#)
- [Configuring Router CE2 on page 439](#)
- [Verifying the VPN Operation on page 440](#)

### Configuring Router CE1

#### Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE1 and Router PE1. Specify the **inet** address family type.

```
[edit interfaces fe-0/0/1.0]
family inet {
  address 18.18.18.1/30;
}
```

2. On Router CE1, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```
[edit interfaces lo0]
unit 0 {
  family inet {
    address 1.1.1.1/32;
  }
}
```

3. On Router CE1, configure an IGP. Include the logical interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1. The IGP can be a static route, RIP, OSPF, ISIS, or EBGp. In this example we configure OSPF.

```
[edit protocols]
ospf {
  area 0.0.0.2 {
    interface fe-0/0/1.0;
    interface lo0.0;
  }
}
```

### Configuring Router PE1

#### Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
so-0/2/0 {
  unit 0 {
    family inet {
      address 19.19.19.1/30;
    }
    family mpls;
  }
}
```

```

    }
  }
  fe-1/2/3 {
    unit 0 {
      family inet {
        address 18.18.18.2/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 2.2.2.2/32;
      }
    }
  }
}

```

2. On Router PE1, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing Fast Ethernet interface and specify the export policy to export BGP routes into OSPF.

```

[edit routing-instances]
vpn2CE1 {
  instance-type vrf;
  interface fe-1/2/3.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    ospf {
      export bgp-to-ospf;
      area 0.0.0.2 {
        interface fe-1/2/3.0;
      }
    }
  }
}

```

3. On Router PE1, configure the RSVP and MPLS protocols to support the label-switched path (LSP). Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure a BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the **inet-vpn** address family and **unicast** traffic type to enable BGP to carry IPv4 network layer reachability information (NLRI) for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE1.

```

[edit protocols]
rsvp {
  interface so-0/2/0.0;
}

```

```

    interface lo0.0;
  }
  mpls {
    label-switched-path To-ASBR1 {
      to 4.4.4.4;
    }
    interface so-0/2/0.0;
    interface lo0.0;
  }
  bgp {
    group To_ASBR1 {
      type internal;
      local-address 2.2.2.2;
      neighbor 4.4.4.4 {
        family inet-vpn {
          unicast;
        }
      }
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-0/2/0.0;
      interface lo0.0;
    }
  }
}

```

4. On Router PE1, configure the BGP local autonomous system number.

```

[edit routing-options]
autonomous-system 100;

```

5. On Router PE1, configure a policy to export the BGP routes into OSPF.

```

[edit policy-options]
policy-statement bgp-to-ospf {
  term 1 {
    from protocol bgp;
    then accept;
  }
  term 2 {
    then reject;
  }
}

```

6. On Router PE1, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```

[edit policy-options]
policy-statement vpnexport {
  term 1 {
    from protocol ospf;
    then {
      community add test_comm;
      accept;
    }
  }
}

```

```

    term 2 {
      then reject;
    }
  }
}

```

7. On Router PE1, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```

[edit policy-options]
policy-statement vpnimport {
  term 1 {
    from {
      protocol bgp;
      community test_comm;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}

```

8. On Router PE1, define the **test\_comm** BGP community with a route target.

```

[edit policy-options]
community test_comm members target:1:100;

```

### Configuring Router P1

#### Step-by-Step Procedure

1. On Router P1, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
so-0/2/1 {
  unit 0 {
    family inet {
      address 19.19.19.2/30;
    }
    family mpls;
  }
}
ge-1/3/0 {
  unit 0 {
    family inet {
      address 20.20.20.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 3.3.3.3/32;
    }
  }
}

```

```
}
}
```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
  interface so-0/2/1.0;
  interface ge-1/3/0.0;
  interface lo0.0;
}
mpls {
  interface lo0.0;
  interface ge-1/3/0.0;
  interface so-0/2/1.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-1/3/0.0;
    interface so-0/2/1.0;
    interface lo0.0;
  }
}
```

### Configuring Router ASBR1

#### Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** addresses families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```
[edit interfaces]
ge-0/0/0 {
  unit 0 {
    family inet {
      address 20.20.20.2/30;
    }
    family mpls;
  }
}
ge-0/1/1 {
  unit 0 {
    family inet {
      address 21.21.21.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
```

```

        family inet {
            address 4.4.4.4/32;
        }
    }
}

```

2. On Router ASBR1, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces and the **lo0.0** logical loopback interface.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface ge-0/0/0.0;
    interface lo0.0;
}
mpls {
    label-switched-path To_PE1 {
        to 2.2.2.2;
    }
    interface lo0.0;
    interface ge-0/0/0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0;
    }
}

```

3. On Router ASBR1, create the **To-PE1** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the **lo0.0** interface address of Router PE1.

```

[edit protocols]
bgp {
    group To-PE1 {
        type internal;
        local-address 4.4.4.4;
        neighbor 2.2.2.2 {
            family inet-vpn {
                unicast;
            }
        }
    }
}

```

4. On Router ASBR1, create the **To-ASBR2** external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address of Router ASBR2.

```

[edit protocols]
bgp {
    group To-ASBR2 {

```

```

    type external;
    family inet-vpn {
        unicast;
    }
    neighbor 21.21.21.2 {
        peer-as 200;
    }
}

```

### Configuring Router ASBR2

#### Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
ge-0/1/1 {
    unit 0 {
        family inet {
            address 21.21.21.2/30;
        }
        family mpls;
    }
}
ge-0/2/3 {
    unit 0 {
        family inet {
            address 22.22.22.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 5.5.5.5/32;
        }
    }
}

```

2. On Router ASBR2, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface ge-0/2/3.0;
    interface lo0.0;
}
mpls {
    label-switched-path To_PE2 {

```

```

        to 7.7.7.7;
    }
    interface lo0.0;
    interface ge-0/2/3.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/3.0;
        interface lo0.0;
    }
}

```

3. On Router ASBR2, create the **To-PE2** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the **lo0.0** interface address of Router PE2.

```

[edit protocols]
bgp {
    group To-PE2 {
        type internal;
        local-address 5.5.5.5;
        neighbor 7.7.7.7 {
            family inet-vpn {
                unicast;
            }
        }
    }
}

```

4. On Router ASBR2, create the **To-ASBR1** external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface on Router ASBR1.

```

[edit protocols]
bgp {
    group To-ASBR1 {
        type external;
        family inet-vpn {
            unicast;
        }
        neighbor 21.21.21.1 {
            peer-as 100;
        }
    }
}

```

### Configuring Router P2

#### Step-by-Step Procedure

1. On Router P2, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** addresses families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
so-0/0/0 {
    unit 0 {

```

```

        family inet {
            address 23.23.23.1/30;
        }
        family mpls;
    }
}
ge-0/2/2 {
    unit 0 {
        family inet {
            address 22.22.22.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 6.6.6.6/32;
        }
    }
}
}

```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface so-0/0/0.0;
    interface ge-0/2/2.0;
    interface lo0.0;
}
mpls {
    interface lo0.0;
    interface ge-0/2/2.0;
    interface so-0/0/0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/2.0;
        interface so-0/0/0.0;
        interface lo0.0;
    }
}

```

### Configuring Router PE2

#### Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
```

```

so-0/0/1 {
  unit 0 {
    family inet {
      address 23.23.23.2/30;
    }
    family mpls;
  }
}
fe-0/3/1 {
  unit 0 {
    family inet {
      address 24.24.24.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 7.7.7.7/32;
    }
  }
}

```

2. On Router PE2, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the BGP peer group within the VRF. Specify AS **20** as the peer AS and specify the IP address of the Fast Ethernet interface on Router CE1 as the neighbor address.

```

[edit routing-instances]
vpn2CE2 {
  instance-type vrf;
  interface fe-0/3/1.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    bgp {
      group To_CE2 {
        peer-as 20;
        neighbor 24.24.24.2;
      }
    }
  }
}

```

3. On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure a BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2. Specify the **inet-vpn** address family and **unicast** traffic type to enable BGP to carry IPv4

NLRI for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and the logical loopback interface on Router PE2.

```
[edit protocols]
rsvp {
  interface so-0/0/1.0;
  interface lo0.0;
}
mpls {
  label-switched-path To-ASBR2 {
    to 5.5.5.5;
  }
  interface so-0/0/1.0;
  interface lo0.0;
}
bgp {
  group To_ASBR2 {
    type internal;
    local-address 7.7.7.7;
    neighbor 5.5.5.5 {
      family inet-vpn {
        unicast;
      }
    }
  }
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface so-0/0/1.0;
    interface lo0.0;
  }
}
```

4. On Router PE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

5. On Router PE2, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
  term 1 {
    from protocol bgp;
    then {
      community add test_comm;
      accept;
    }
  }
  term 2 {
    then reject;
  }
}
```

- On Router PE2, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```
[edit policy-options]
policy-statement vpnimport {
  term 1 {
    from {
      protocol bgp;
      community test_comm;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

- On Router PE1, define the **test\_comm** BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

### Configuring Router CE2

#### Step-by-Step Procedure

- On Router CE2, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE2 and Router PE2. Specify the **inet** address family type.

```
[edit interfaces]
fe-3/0/0 {
  unit 0 {
    family inet {
      address 24.24.24.2/30;
    }
  }
}
```

- On Router CE2, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```
[edit interfaces]
lo0 {
  unit 0 {
    family inet {
      address 8.8.8.8/32;
    }
  }
}
```

- On Router CE2, configure an IGP. The IGP can be a static route, RIP, OSPF, ISIS, or EBGp. In this example, we configure EBGp. Specify AS **200** as the peer AS and specify the BGP neighbor IP address as the Fast Ethernet interface of Router PE2. Include the **export** statement.

```
[edit protocols]
bgp {
  group To_PE2 {
```

```

neighbor 24.24.24.1 {
  export myroutes;
  peer-as 200;
}
}
}

```

## Verifying the VPN Operation

### Step-by-Step Procedure

1. Commit the configuration on each router.



**NOTE:** The MPLS labels shown in this example will be different than the labels used in your configuration.

2. On Router PE1, display the routes for the **vpn2CE1** routing instance using the **show ospf route** command. Verify that the 1.1.1.1 route is learned from OSPF.

```

user@PE1> show ospf route instance vpn2CE1
Topology default Route Table:

```

Prefix	Path	Route	NH	Metric	NextHop	Nexthop
	Type	Type	Type		Interface	addr/label
1.1.1.1	Intra	Router	IP	1	fe-1/2/3.0	18.18.18.1
1.1.1.1/32	Intra	Network	IP	1	fe-1/2/3.0	18.18.18.1

3. On Router PE1, use the **show route advertising-protocol** command to verify that Router PE1 advertises the 1.1.1.1 route to Router ASBR1 using MP-BGP with the VPN MPLS label.

```

user@PE1> show route advertising-protocol bgp 4.4.4.4 extensive
vpn2CE1.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  BGP group To_ASBR1 type Internal
    Route Distinguisher: 1:100
    VPN Label: 299952
    Nexthop: Self
    Flags: Nexthop Change
    MED: 1
    Localpref: 100
    AS path: [100] I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

4. On Router ASBR1, use the **show route receive-protocol** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **bgp.l3vpn.0** routing table.

```

user@ASBR1> show route receive-protocol bgp 2.2.2.2 extensive
inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:1.1.1/32 (1 entry, 1 announced)

```

```

Route Distinguisher: 1:100
VPN Label: 299952
Nexthop: 2.2.2.2
MED: 1
Localpref: 100
AS path: I
Communities: target:1:100 rte-type:0.0.0.2:1:0

```

5. On Router ASBR1, use the **show route advertising-protocol** command to verify that Router ASBR1 advertises the 1.1.1.1 route to Router ASBR2.

```

user@ASBR1> show route advertising-protocol bgp 21.21.21.2 extensive
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:1.1.1/32 (1 entry, 1 announced)
  BGP group To-ASBR2 type External
    Route Distinguisher: 1:100
    VPN Label: 299984
    Nexthop: Self
    Flags: Nexthop Change
    AS path: [100] I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

6. On Router ASBR2, use the **show route receive-protocol** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **bgp.l3vpn.0** routing table.

```

user@ASBR2> show route receive-protocol bgp 21.21.21.1 extensive
inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:1.1.1/32 (1 entry, 1 announced)
  Accepted
    Route Distinguisher: 1:100
    VPN Label: 299984
    Nexthop: 21.21.21.1
    AS path: 100 I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

7. On Router ASBR2, use the **show route advertising-protocol** command to verify that Router ASBR2 advertises the 1.1.1.1 route to Router PE2 in the **To-PE2** routing instance.

```

user@ASBR2> show route advertising-protocol bgp 7.7.7.7 extensive
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:1.1.1/32 (1 entry, 1 announced)
  BGP group To-PE2 type Internal
    Route Distinguisher: 1:100
    VPN Label: 300048
    Nexthop: Self
    Flags: Nexthop Change
    Localpref: 100
    AS path: [200] 100 I
    Communities: target:1:100 rte-type:0.0.0.2:1:0

```

8. On Router PE2, use the **show route receive-protocol** command to verify that the router receives and accepts the 1.1.1.1 route and places it in the **To\_CE2.inet.0** routing table.

```

user@PE2> show route receive-protocol bgp 5.5.5.5 extensive
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

__juniper_private1__.inet.0: 14 destinations, 14 routes (8 active, 0 holddown,
6 hidden)

__juniper_private2__.inet.0: 1 destinations, 1 routes (0 active, 0 holddown,
1 hidden)

To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  Accepted
  Route Distinguisher: 1:100
  VPN Label: 300048
  Nexthop: 5.5.5.5
  Localpref: 100
  AS path: 100 I
  AS path: Recorded
  Communities: target:1:100 rte-type:0.0.0.2:1:0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 1:100:1.1.1.1/32 (1 entry, 0 announced)
  Accepted
  Route Distinguisher: 1:100
  VPN Label: 300048
  Nexthop: 5.5.5.5
  Localpref: 100
  AS path: 100 I
  AS path: Recorded
  Communities: target:1:100 rte-type:0.0.0.2:1:0

__juniper_private1__.inet6.0: 4 destinations, 4 routes (4 active, 0 holddown,
0 hidden)

```

9. On Router PE2, use the **show route advertising-protocol** command to verify that Router PE2 advertises the 1.1.1.1 route to Router CE2 through the **To\_CE2** peer group.

```

user@PE2> show route advertising-protocol bgp 24.24.24.2 extensive
To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  BGP group To_CE2 type External
  Nexthop: Self
  AS path: [200] 100 I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

10. On Router CE2, use the **show route** command to verify that Router CE2 receives the 1.1.1.1 route from Router PE2.

```

user@CE2> show route 1.1.1.1
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32          *[BGP/170] 00:25:36, localpref 100
                    AS path: 200 100 I
                    > to 24.24.24.1 via fe-3/0/0.0

```

11. On Router CE2, use the **ping** command and specify **8.8.8.8** as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 1.1.1.1 source 8.8.8.8
PING 1.1.1.1 (1.1.1.1): 56 data bytes
64 bytes from 1.1.1.1: icmp_seq=0 ttl=58 time=4.786 ms
64 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=10.210 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=10.588 ms
```

12. On Router PE2, use the **show route** command to verify that the traffic is sent with an inner label of **300048** and a top label of **299776**.

```
user@PE2> show route 1.1.1.1 detail
To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
1.1.1.1/32 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Route Distinguisher: 1:100
            Next hop type: Indirect
            Next-hop reference count: 3
            Source: 5.5.5.5
            Next hop type: Router, Next hop index: 653
            Next hop: via so-0/0/1.0 weight 0x1, selected
            Label-switched-path To-PE2
            Label operation: Push 300048, Push 299776(top)
            Protocol next hop: 5.5.5.5
            Push 300048
            Indirect next hop: 8c61138 262143
            State: <Secondary Active Int Ext>
            Local AS: 200 Peer AS: 200
            Age: 27:48 Metric2: 2
            Task: bgp_200.5.5.5+60185
            Announcement bits (3): 0-RT 1-KRT 2-BGP RT Background
            AS path: 100 I
            AS path: Recorded
            Communities: target:1:100 rte-type:0.0.0.2:1:0
            Accepted
            VPN Label: 300048
            Localpref: 100
            Router ID: 5.5.5.5
            Primary Routing Table bgp.l3vpn.0
```

13. On Router ASBR2, use the **show route table** command to verify that Router ASBR2 receives the traffic after the top label is popped by Router P2, that label **300048** is swapped with label **299984**, and that the packet is sent toward Router ASBR1 through interface **ge-0/1/1.0**.

```
user@ASBR2> show route table mpls.0 detail
300048 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Router, Next hop index: 648
            Next-hop reference count: 2
            Source: 21.21.21.1
            Next hop: 21.21.21.1 via ge-0/1/1.0, selected
            Label operation: Swap 299984
            State: <Active Int Ext>
            Local AS: 200
            Age: 30:39
            Task: BGP RT Background
            Announcement bits (1): 0-KRT
            AS path: 100 I
```

```
Ref Cnt: 1
Communities: target:1:100 rte-type:0.0.0.2:1:0
```

14. On Router ASBR1, use the **show route table** command to verify that Router ASBR1 receives the traffic with label **299984**, swaps the label with **299952**, and pushes a new top label of **299792**.

```
user@ASBR1> show route table mpls.0 detail
299984 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Indirect
            Next-hop reference count: 2
            Source: 2.2.2.2
            Next hop type: Router, Next hop index: 538
            Next hop: 20.20.20.1 via ge-0/0/0.0 weight 0x1, selected
            Label-switched-path To_PE1
            Label operation: Swap 299952, Push 299792(top)
            Protocol next hop: 2.2.2.2
            Swap 299952
            Indirect next hop: 8af70a0 262142
            State: <Active Int Ext>
            Local AS: 100
            Age: 34:09      Metric2: 2
            Task: BGP RT Background
            Announcement bits (1): 0-KRT
            AS path: I
            Ref Cnt: 1
            Communities: target:1:100 rte-type:0.0.0.2:1:0
```

15. On Router PE1, use the **show route table** command to verify that Router PE1 receives the traffic with label **299952**, and then pops the inner label.

```
user@PE1> show route table mpls.0 detail
299952 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Router, Next hop index: 536
            Next-hop reference count: 2
            Next hop: 18.18.18.1 via fe-1/2/3.0, selected
            Label operation: Pop
            State: Active Int Ext
            Local AS: 100
            Age: 40:26
            Task: BGP RT Background
            Announcement bits (1): 0-KRT
            AS path: I
            Ref Cnt: 1
            Communities: rte-type:0.0.0.2:1:0
```

## Related Documentation

### Example: Configuring Interprovider Layer 3 VPN Option C

Interprovider Layer 3 VPN Option C provides interprovider multihop EBGp redistribution of labeled VPN-IPv4 routes between source and destination ASs, with EBGp redistribution of labeled IPv4 routes from AS to neighboring AS. Compared to Option A and Option B, Option C is the most scalable solution. To configure an interprovider Layer 3 VPN option

C service, you need to configure the AS border routers and the PE routers connected to the end customer's CE routers using multihop EBGp.

This example provides a step-by-step procedure to configure interprovider layer 3 VPN option C, which is one of the recommended implementations of MPLS VPN when that service is required by a customer that has more than one AS but not all of the customer's ASs can be serviced by the same service provider (SP). It is organized in the following sections:

- [Requirements on page 445](#)
- [Configuration Overview and Topology on page 445](#)
- [Configuration on page 447](#)

## Requirements

This example requires the following hardware and software components:

- Junos OS Release 9.5 or later.
- Eight Juniper Networks M Series Multiservice Edge Routers, T Series Core Routers, TX Matrix Routers, or MX Series 3D Universal Edge Routers.

## Configuration Overview and Topology

Interprovider layer 3 VPN option C is a very scalable interprovider VPN solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same SP.

*RFC 4364* section 10, refers to this method as multihop EBGp redistribution of labeled VPN-IPv4 routes between source and destination ASs, with EBGp redistribution of labeled IPv4 routes from AS to neighboring AS.

This solution is similar to the solution described in *Implementing Interprovider Layer 3 VPN Option B*, except internal IPv4 unicast routes are advertised instead of external VPN-IPv4-unicast routes, using EBGp. Internal routes are internal to leaf SPs (SP1 and SP2 in this example), and external routes are those learned from the end customer requesting VPN services.

In this configuration:

- After the loopback address of Router PE2 is learned by Router PE1 and the loopback address of Router PE1 is learned by Router PE2, the end PE routers establish an MP-EBGP session for exchanging VPN-IPv4 routes.
- Since VPN-IPv4 routes are exchanged among end PE routers, any other router on the path from Router PE1 and Router PE2 does not need to keep or install VPN-IPv4 routes in their routing information base (RIB) or forwarding information base (FIB) tables.
- An MPLS path needs to be established between Router PE1 and Router PE2.

*RFC 4364* describes only one solution that uses a BGP labeled-unicast approach. In this approach, the ASBR routers advertise the loopback addresses of the PE routers and

associate each prefix with a label according to *RFC 3107*. Service providers may use RSVP or LDP to establish an LSP between ASBR routers and PE routers in their internal network.

In this network, ASBR2 receives label information associated with the loopback IP address of Router PE1 and advertises another label to Router ASBR1 using MP-EBGP labeled-unicast. Meanwhile, the ASBRs build their own MPLS forwarding table according to the received and advertised routes and labels. Router ASBR1 uses its own IP address as the next-hop information.

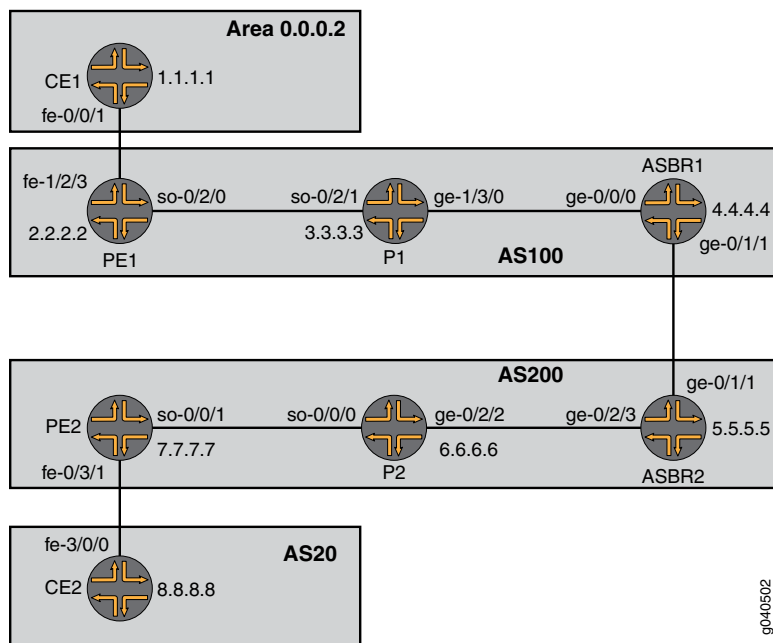
Router ASBR2 receives this prefix associated with a label, assigns another label, changes the next-hop address to its own address, and advertises it to Router PE1. Router PE1 now has an update with the label information and next-hop to Router ASBR1. Also, Router PE1 already has a label associated with the IP address of Router ASBR1. If Router PE1 sends an IP packet to Router PE2, it pushes two labels: one for the IP address of Router PE2 (obtained using MP-IBGP labeled-unicast advertisement) and one for the IP address of Router ASBR1 (obtained using LDP or RSVP).

Router ASBR1 then pops the outer label and swaps the inner label with the label learned from a neighbor ASBR for its neighboring PE router. Router ASBR2 performs a similar function and swaps the incoming label (only one) and pushes another label that is associated with the address of Router PE2. Router PE2 pops both labels and passes the remaining IP packet to its own CPU. After the end-to-end connection among the PE routers is created, the PE routers establish an MP-EBGP session to exchange VPN-IPv4 routes.

In this solution, PE routers push three labels onto the IP packet coming from the VPN end user. The inner-most label, obtained using MP-EBGP, determines the correct VPN routing and forwarding (VRF) routing instance at the remote PE. The middle label is associated with the IP address of the remote PE and is obtained from an ASBR using MP-IBGP labeled-unicast. The outer label is associated with the IP addresses of the ASBRs and is obtained using LDP or RSVP.

The physical topology of the network is shown in [Figure 57 on page 447](#).

Figure 57: Physical Topology of Interprovider Layer 3 VPN Option C



## Configuration



**NOTE:** The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure interprovider layer 3 VPN option C, perform the following tasks:

- [Configuring Router CE1 on page 447](#)
- [Configuring Router PE1 on page 448](#)
- [Configuring Router P1 on page 451](#)
- [Configuring Router ASBR1 on page 452](#)
- [Configuring Router ASBR2 on page 455](#)
- [Configuring Router P2 on page 457](#)
- [Configuring Router PE2 on page 458](#)
- [Configuring Router CE2 on page 461](#)
- [Verifying the VPN Operation on page 462](#)

### Configuring Router CE1

#### Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE1 and Router PE1. Specify the **inet** address family type.

```
[edit interfaces fe-0/0/1.0]
family inet {
  address 18.18.18.1/30;
}
```

2. On Router CE1, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```
[edit interfaces lo0]
unit 0 {
  family inet {
    address 1.1.1.1/32;
  }
}
```

3. On Router CE1, configure an IGP. The IGP can be a static route, RIP, OSPF, ISIS, or EBGp. In this example we configure OSPF. Include the logical interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1.

```
[edit protocols]
ospf {
  area 0.0.0.2 {
    interface fe-0/0/1.0;
    interface lo0.0 {
      passive;
    }
  }
}
```

### Configuring Router PE1

#### Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET interfaces.

```
[edit interfaces]
so-0/2/0 {
  unit 0 {
    family inet {
      address 19.19.19.1/30;
    }
    family mpls;
  }
}
fe-1/2/3 {
  unit 0 {
    family inet {
      address 18.18.18.2/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 2.2.2.2/32;
    }
  }
}
```

```

    }
  }
}

```

2. On Router PE1, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing Fast Ethernet interface and specify the export policy to export BGP routes into OSPF.

```

[edit routing-instances]
vpn2CE1 {
  instance-type vrf;
  interface fe-1/2/3.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    ospf {
      export bgp-to-ospf;
      area 0.0.0.2 {
        interface fe-1/2/3.0;
      }
    }
  }
}

```

3. On Router PE1, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE1.

```

[edit protocols]
rsvp {
  interface so-0/2/0.0;
  interface lo0.0;
}
mpls {
  label-switched-path To-ASBR1 {
    to 4.4.4.4;
  }
  interface so-0/2/0.0;
  interface lo0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface so-0/2/0.0;
    interface lo0.0 {
      passive;
    }
  }
}

```

4. On Router PE1, configure the **To\_ASBR1** peer BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the **inet** address family. For a PE router to install a route in the VRF, the next hop must resolve to a route stored within the **inet.3** table. The **labeled-unicast resolve-vpn** statements allow labeled routes to be placed in the **inet.3** routing table for route resolution, which are then resolved for PE router connections where the remote PE is located across another AS.

```
[edit protocols]
bgp {
  group To_ASBR1 {
    type internal;
    local-address 2.2.2.2;
    neighbor 4.4.4.4 {
      family inet {
        labeled-unicast {
          resolve-vpn;
        }
      }
    }
  }
}
```

5. On Router PE1, configure multihop EBGp toward PE2. Specify the **inet-vpn** family.

```
[edit protocols]
bgp {
  group To_PE2 {
    multihop {
      ttl 20;
    }
    local-address 2.2.2.2;
    family inet-VPN {
      unicast;
    }
    neighbor 7.7.7.7 {
      peer-as 200;
    }
  }
}
```

6. On Router PE1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

7. On Router PE1, configure a policy to export the BGP routes into OSPF.

```
[edit policy-options]
policy-statement bgp-to-ospf {
  term 1 {
    from protocol bgp;
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

```
}
}
```

8. On Router PE1, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
  term 1 {
    from protocol ospf;
    then {
      community add test_comm;
      accept;
    }
  }
  term 2 {
    then reject;
  }
}
```

9. On Router PE1, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```
[edit policy-options]
policy-statement vpnimport {
  term 1 {
    from {
      protocol bgp;
      community test_comm;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

10. On Router PE1, define the **test\_comm** BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

### Configuring Router P1

#### Step-by-Step Procedure

1. On Router P1, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```
[edit interfaces]
so-0/2/1 {
  unit 0 {
    family inet {
      address 19.19.19.2/30;
    }
    family mpls;
  }
}
```

```

}
ge-1/3/0 {
  unit 0 {
    family inet {
      address 20.20.20.1/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 3.3.3.3/32;
    }
  }
}
}

```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
  interface so-0/2/1.0;
  interface ge-1/3/0.0;
  interface lo0.0;
}
mpls {
  interface lo0.0;
  interface ge-1/3/0.0;
  interface so-0/2/1.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-1/3/0.0;
    interface so-0/2/1.0;
    interface lo0.0 {
      passive;
    }
  }
}
}

```

### Configuring Router ASBR1

#### Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** addresses families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
ge-0/0/0 {
  unit 0 {

```

```

        family inet {
            address 20.20.20.2/30;
        }
        family mpls;
    }
}
ge-0/1/1 {
    unit 0 {
        family inet {
            address 21.21.21.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 4.4.4.4/32;
        }
    }
}
}

```

2. On Router ASBR1, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces and the logical loopback interface. Include the **traffic-engineering bgp-igp-both-ribs** statement at the **[edit protocols mpls]** hierarchy level.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface ge-0/0/0.0;
    interface lo0.0;
}
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path To_PE1 {
        to 2.2.2.2;
    }
    interface lo0.0;
    interface ge-0/0/0.0;
    interface ge-0/1/1.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
}

```

3. On Router ASBR1, create the **To-PE1** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the Gigabit Ethernet interface address of Router PE1.

```
[edit protocols]
bgp {
  group To-PE1 {
    type internal;
    local-address 4.4.4.4;
    neighbor 2.2.2.2 {
      family inet {
        labeled-unicast;
      }
      export next-hop-self;
    }
  }
}
```

4. On Router ASBR1, create the **To-ASBR2** external BGP peer group. Enable the router to use BGP to advertise network layer reachability information (NLRI) for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address on Router ASBR2.

```
[edit protocols]
group To-ASBR2 {
  type external;
  family inet {
    labeled-unicast;
  }
  export To-ASBR2;
  neighbor 21.21.21.2 {
    peer-as 200;
  }
}
```

5. On Router ASBR1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

6. On Router ASBR 1, configure a policy to import routes from BGP that match the 2.2.2.2/32 route.

```
[edit policy-options]
policy-statement To-ASBR2 {
  term 1 {
    from {
      route-filter 2.2.2.2/32 exact;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

7. On Router ASBR 1, define a next-hop self policy and apply it to the IBGP sessions.

```
[edit policy-options]
policy-statement next-hop-self {
```

```

    then {
        next-hop self;
    }
}

```

### Configuring Router ASBR2

#### Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** address families. Configure the IP address for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```

[edit interfaces]
ge-0/1/1 {
    unit 0 {
        family inet {
            address 21.21.21.2/30;
        }
        family mpls;
    }
}
ge-0/2/3 {
    unit 0 {
        family inet {
            address 22.22.22.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 5.5.5.5/32;
        }
    }
}

```

2. On Router ASBR2, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces. Include the **traffic-engineering bgp-igp-both-ribs** statement at the **[edit protocols mpls]** hierarchy level.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```

[edit protocols]
rsvp {
    interface ge-0/2/3.0;
    interface lo0.0;
}
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path To_PE2 {
        to 7.7.7.7;
    }
}

```

```

interface lo0.0
interface ge-0/2/3.0;
interface ge-0/1/1.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-0/2/3.0;
    interface lo0.0 {
      passive;
    }
  }
}
}

```

3. On Router ASBR2, create the **To-PE2** internal BGP peer group. Specify the local IP peer address as the local **lo0.0** address. Specify the neighbor IP peer address as the **lo0.0** interface address of Router PE2.

```

[edit protocols]
bgp {
  group To-PE2 {
    type internal;
    local-address 5.5.5.5;
    export next-hop-self;
    neighbor 7.7.7.7 {
      family inet {
        labeled-unicast;
      }
      export next-hop-self;
    }
  }
}

```

4. On Router ASBR2, create the **To-ASBR1** external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address on Router ASBR1.

```

[edit protocols]
bgp {
  group To-ASBR1 {
    type external;
    family inet {
      labeled-unicast;
    }
    export To-ASBR1;
    neighbor 21.21.21.1 {
      peer-as 100;
    }
  }
}

```

5. On Router ASBR2 configure the BGP local autonomous system number.

```

[edit routing-options]
autonomous-system 200;

```

6. On Router ASBR2, configure a policy to import routes from BGP that match the 7.7.7.7/32 route.

```
[edit policy-options]
policy-statement To-ASBR1 {
  term 1 {
    from {
      route-filter 7.7.7/32 exact;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

7. On Router ASBR 2, define a next-hop self policy.

```
[edit policy-options]
policy-statement next-hop-self {
  then {
    next-hop self;
  }
}
```

### Configuring Router P2

#### Step-by-Step Procedure

1. On Router P2, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the **inet** and **mpls** addresses families. Configure the IP addresses for the **lo0.0** loopback interface and enable the interface to process the **inet** address family.

```
[edit interfaces]
so-0/0/0 {
  unit 0 {
    family inet {
      address 23.23.23.1/30;
    }
    family mpls;
  }
}
ge-0/2/2 {
  unit 0 {
    family inet {
      address 22.22.22.2/30;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 6.6.6.6/32;
    }
  }
}
```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
  interface so-0/0/0.0;
  interface ge-0/2/2.0;
  interface lo0.0;
}
mpls {
  interface lo0.0;
  interface ge-0/2/2.0;
  interface so-0/0/0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface ge-0/2/2.0;
    interface so-0/0/0.0;
    interface lo0.0 {
      passive;
    }
  }
}
```

### Configuring Router PE2

#### Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the **inet** address family on all of the interfaces. Specify the **mpls** address family on the SONET interface.

```
[edit interfaces]
so-0/0/1 {
  unit 0 {
    family inet {
      address 23.23.23.2/30;
    }
    family mpls;
  }
}
fe-0/3/1 {
  unit 0 {
    family inet {
      address 24.24.24.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 7.7.7.7/32;
    }
  }
}
```

```

    }
  }
}

```

- On Router PE2, configure the routing instance for VPN2. Specify the **vrf** instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the BGP peer group within the VRF. Specify AS **20** as the peer AS and specify the IP address of the Fast Ethernet interface on Router CE1 as the neighbor address.

```

[edit routing-instances]
vpn2CE2 {
  instance-type vrf;
  interface fe-0/3/1.0;
  route-distinguisher 1:100;
  vrf-import vpnimport;
  vrf-export vpnexport;
  protocols {
    bgp {
      group To_CE2 {
        peer-as 20;
        neighbor 24.24.24.2;
      }
    }
  }
}

```

- On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE2.

```

[edit protocols]
rsvp {
  interface so-0/0/1.0;
  interface lo0.0;
}
mpls {
  label-switched-path To-ASBR2 {
    to 5.5.5.5;
  }
  interface so-0/0/1.0;
  interface lo0.0;
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface so-0/0/1.0;
    interface lo0.0 {
      passive;
    }
  }
}

```

4. On Router PE2, configure the **To\_ASBR2** BGP group. Specify the group type as **internal**. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2.

```
[edit protocols]
bgp {
  group To_ASBR2 {
    type internal;
    local-address 7.7.7.7;
    neighbor 5.5.5.5 {
      family inet {
        labeled-unicast {
          resolve-vpn;
        }
      }
    }
  }
}
```

5. On Router PE2, configure multihop EBGP towards Router PE1. Specify the **inet-vpn** address family.

```
[edit protocols]
bgp {
  group To_PE1 {
    type external;
    local-address 7.7.7.7;
    multihop {
      ttl 20;
    }
    family inet-vpn {
      unicast;
    }
    neighbor 2.2.2.2 {
      peer-as 100;
    }
  }
}
```

6. On Router PE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

7. On Router PE2, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
  term 1 {
    from protocol bgp;
    then {
      community add test_comm;
      accept;
    }
  }
  term 2 {
    then reject;
  }
}
```

```
}
}
```

8. On Router PE2, configure a policy to import routes from BGP that have the **test\_comm** community attached.

```
[edit policy-options]
policy-statement vpnimport {
  term 1 {
    from {
      protocol bgp;
      community test_comm;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

9. On Router PE1, define the **test\_comm** BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

### Configuring Router CE2

#### Step-by-Step Procedure

1. On Router CE2, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE2 and Router PE2. Specify the **inet** address family type.

```
[edit interfaces]
fe-3/0/0 {
  unit 0 {
    family inet {
      address 24.24.24.2/30;
    }
  }
}
```

2. On Router CE2, configure the IP address and protocol family on the loopback interface. Specify the **inet** address family type.

```
[edit interfaces lo0]
lo0 {
  unit 0 {
    family inet {
      address 8.8.8.8/32;
    }
  }
}
```

3. On Router CE2, define a policy named **myroutes** that accepts direct routes.

```
[edit policy-options]
policy-statement myroutes {
  from protocol direct;
  then accept;
```

```
}
```

4. On Router CE2, configure an IGP. The IGP can be a static route, RIP, OSPF, ISIS, or EBGP. In this example, we configure EBGP. Specify the BGP neighbor IP address as the logical loopback interface of Router PE1. Apply the **myroutes** policy.

```
[edit protocols]
bgp {
  group To_PE2 {
    neighbor 24.24.24.1 {
      export myroutes;
      peer-as 200;
    }
  }
}
```

5. On Router CE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 20;
```

### Verifying the VPN Operation

#### Step-by-Step Procedure

1. Commit the configuration on each router.



**NOTE:** The MPLS labels shown in this example will be different than the labels used in your configuration.

2. On Router PE1, display the routes for the **vpn2CE1** routing instance using the **show ospf route** command. Verify that the 1.1.1.1 route is learned from OSPF.

```
user@PE1> show ospf route instance vpn2CE1
Topology default Route Table:
```

Prefix	Path	Route	NH	Metric	NextHop	Nexthop
	Type	Type	Type		Interface	addr/label
1.1.1.1	Intra	Router	IP	1	fe-1/2/3.0	18.18.18.1
1.1.1.1/32	Intra	Network	IP	1	fe-1/2/3.0	18.18.18.1
18.18.18.0/30	Intra	Network	IP	1	fe-1/2/3.0	

3. On Router PE1, use the **show route advertising-protocol bgp 7.7.7 extensive** command to verify that Router PE1 advertises the 1.1.1.1 route to Router PE2 using MP-BGP with the VPN MPLS label.

```
user@PE1> show route advertising-protocol bgp 7.7.7 extensive
bgp.13vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:1.1.1.1/32 (1 entry, 1 announced)
BGP group To_PE2 type External
  Route Distinguisher: 1:100
  VPN Label: 300016
  Nexthop: Self
  Flags: Nexthop Change
  MED: 1
  AS path: [100] I
  Communities: target:1:100 rte-type:0.0.0.2:1:0
```

4. On Router ASBR1, use the **show route advertising-protocol** command to verify that Router ASBR1 advertises the **2.2.2.2** route to Router ASBR2.

```
user@ASBR1> show route advertising-protocol bgp 21.21.21.2 extensive
inet.0: 14 destinations, 16 routes (14 active, 0 holddown, 0 hidden)
* 2.2.2.2/32 (2 entries, 1 announced)
  BGP group To-PE2 type External
    Route Label: 300172
    Nexthop: Self
    Flags: Nexthop Change
    MED: 2
    AS path: [100] I
```

5. On Router ASBR2, use the **show route receive-protocol** command to verify that the router receives and accepts the **2.2.2.2** route and places it in the **To\_ASBR2.inet.0** routing table.

```
user@ASBR2> show route receive-protocol bgp 21.21.21.1 extensive
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
* 2.2.2.2/32 (1 entry, 1 announced)
  Accepted
    Route Label: 300172
    Nexthop: 21.21.21.1
    MED: 2
    AS path: 100 I
```

6. On Router ASBR2, use the **show route advertising-protocol** command to verify that Router ASBR2 advertises the **2.2.2.2** route to Router PE2 in the **To-PE2** routing instance.

```
user@ASBR2> show route advertising-protocol bgp 7.7.7.7 extensive
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
* 2.2.2.2/32 (1 entry, 1 announced)
  BGP group To-PE2 type Internal
    Route Label: 300192
    Nexthop: Self
    Flags: Nexthop Change
    MED: 2
    Localpref: 100
    AS path: [200] 100 I
```

7. On Router PE2, use the **show route receive-protocol** command to verify that Router PE2 receives the route and puts it in the **inet.0** routing table. Verify that Router PE2 also receives the update from Router PE1 and accepts the route.

```
user@PE2> show route receive-protocol bgp 5.5.5.5 extensive
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
* 2.2.2.2/32 (1 entry, 1 announced)
  Accepted
    Route Label: 300192
    Nexthop: 5.5.5.5
    MED: 2
    Localpref: 100
    AS path: 100 I
    AS path: Recorded
```

```
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
* 2.2.2.2/32 (1 entry, 1 announced)
  Accepted
    Route Label: 300192
```

```

Nexthop: 5.5.5.5
MED: 2
Localpref: 100
AS path: 100 I
AS path: Recorded

```

8. On Router PE2, use the **show route receive-protocol** command to verify that Router PE2 puts the route in the routing table of the **To\_CE2** routing instance and advertises the route to Router CE2 using EBGp.

```

user@PE2> show route receive-protocol bgp 2.2.2.2 detail
inet.0: 17 destinations, 18 routes (17 active, 0 holddown, 0 hidden)

inet.3: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

__juniper_private1__.inet.0: 14 destinations, 14 routes (8 active, 0 holddown,
6 hidden)

__juniper_private2__.inet.0: 1 destinations, 1 routes (0 active, 0 holddown,
1 hidden)

To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  Accepted
  Route Distinguisher: 1:100
  VPN Label: 300016
  Nexthop: 2.2.2.2
  MED: 1
  AS path: 100 I
  AS path: Recorded
  Communities: target:1:100 rte-type:0.0.0.2:1:0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

* 1:100:1.1.1.1/32 (1 entry, 0 announced)
  Accepted
  Route Distinguisher: 1:100
  VPN Label: 300016
  Nexthop: 2.2.2.2
  MED: 1
  AS path: 100 I
  AS path: Recorded
  Communities: target:1:100 rte-type:0.0.0.2:1:0

__juniper_private1__.inet6.0: 4 destinations, 4 routes (4 active, 0 holddown,
0 hidden)

```

9. On Router PE2, use the **show route advertising-protocol** command to verify that Router PE2 advertises the 1.1.1.1 route to Router CE2 through the **To\_CE2** peer group.

```

user@PE2> show route advertising-protocol bgp 24.24.24.2 extensive
To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 1.1.1.1/32 (1 entry, 1 announced)
  BGP group To_CE2 type External
  Nexthop: Self
  AS path: [200] 100 I
  Communities: target:1:100 rte-type:0.0.0.2:1:0

```

10. On Router CE2, use the **show route** command to verify that Router CE2 receives the 1.1.1.1 route from Router PE2.

```
user@CE2> show route 1.1.1.1
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.1/32          *[BGP/170] 00:25:36, localpref 100
                   AS path: 200 100 I
                   > to 24.24.24.1 via fe-3/0/0.0
```

11. On Router CE2, use the **ping** command and specify 8.8.8.8 as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 1.1.1.1 source 8.8.8.8
PING 1.1.1.1 (1.1.1.1): 56 data bytes
64 bytes from 1.1.1.1: icmp_seq=0 ttl=58 time=4.786 ms
64 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=10.210 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=10.588 ms
```

12. On Router PE2, use the **show route** command to verify that the traffic is sent with an inner label of 300016, a middle label of 300192, and a top label of 299776.

```
user@PE2> show route 1.1.1.1 detail
To_CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
1.1.1.1/32 (1 entry, 1 announced)
   *BGP      Preference: 170/-101
              Route Distinguisher: 1:100
              Next hop type: Indirect
              Next-hop reference count: 3
              Source: 2.2.2.2
              Next hop type: Router, Next hop index: 653
              Next hop: via so-0/0/1.0 weight 0x1, selected
              Label-switched-path To-ASBR2
              Label operation: Push 300016, Push 300192, Push 299776(top)
              Protocol next hop: 2.2.2.2
              Push 300016
              Indirect next hop: 8c61138 262142
              State: <Secondary Active Ext>
              Local AS: 200 Peer AS: 100
              Age: 17:33      Metric: 1      Metric2: 2
              Task: BGP_100.2.2.2+62319
              Announcement bits (3): 0-RT 1-KRT 2-BGP RT Background
              AS path: 100 I
              AS path: Recorded
              Communities: target:1:100 rte-type:0.0.0.2:1:0
              Accepted
              VPN Label: 300016
              Localpref: 100
              Router ID: 2.2.2.2
              Primary Routing Table bgp.l3vpn.0
```

13. On Router ASBR2, use the **show route table** command to verify that Router ASBR2 receives the traffic after the top label is popped by Router P2. Verify that label 300192 is a swapped with label 300176 and the traffic is sent towards Router ASBR1 using interface ge-0/1/1.0. At this point, the bottom label 300016 is preserved.

```
lab@ASBR2# show route table mpls.0 detail
300192 (1 entry, 1 announced)
   *VPN      Preference: 170
              Next hop type: Router, Next hop index: 660
```

```

Next-hop reference count: 2
Source: 21.21.21.1
Next hop: 21.21.21.1 via ge-0/1/1.0, selected
Label operation: Swap 300176
State: <Active Int Ext>
Local AS: 200
Age: 24:01
Task: BGP RT Background
Announcement bits (1): 0-KRT
AS path: 100 I
Ref Cnt: 1

```

14. On Router ASBR1, use the **show route table** command to verify that when Router ASBR1 receives traffic with label **300176**, it swaps the label with **299824** to reach Router PE1.

```

user@ASBR1> show route table mpls.0 detail
300176 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Router, Next hop index: 651
            Next-hop reference count: 2
            Next hop: 20.20.20.1 via ge-0/0/0.0 weight 0x1, selected
            Label operation: Swap 299824
            State: <Active Int Ext>
            Local AS: 100
            Age: 25:53
            Task: BGP RT Background
            Announcement bits (1): 0-KRT
            AS path: I
            Ref Cnt: 1

```

15. On Router PE1, use the **show route table** command to verify that Router PE1 receives the traffic after the top label is popped by Router P1. Verify that label **300016** is popped and the traffic is sent towards Router CE1 using interface **fe-1/2/3.0**.

```

user@PE1> show route table mpls.0 detail
300016 (1 entry, 1 announced)
  *VPN      Preference: 170
            Next hop type: Router, Next hop index: 643
            Next-hop reference count: 2
            Next hop: 18.18.18.1 via fe-1/2/3.0, selected
            Label operation: Pop
            State: <Active Int Ext>
            Local AS: 100
            Age: 27:37
            Task: BGP RT Background
            Announcement bits (1): 0-KRT
            AS path: I
            Ref Cnt: 1
            Communities: rte-type:0.0.0.2:1:0

```

**Related  
Documentation**

## Layer 3 VPN Overview

Layer 3 VPNs are based on RFC 2547bis, *BGP/MPLS IP VPNs*. RFC 2547bis defines a mechanism by which service providers can use their IP backbones to provide VPN services

to their customers. A Layer 3 VPN is a set of sites that share common routing information and whose connectivity is controlled by a collection of policies. The sites that make up a Layer 3 VPN are connected over a provider's existing Internet backbone. RFC 2547bis VPNs are also known as BGP/MPLS VPNs because BGP is used to distribute VPN routing information across the provider's backbone, and MPLS is used to forward VPN traffic across the backbone to remote VPN sites.

Customer networks, because they are private, can use either public addresses or private addresses, as defined in RFC 1918, *Address Allocation for Private Internets*. When customer networks that use private addresses connect to the Internet infrastructure, the private addresses might overlap with the same private addresses used by other network users. MPLS/BGP VPNs solve this problem by adding a *route distinguisher*. A route distinguisher is a VPN identifier prefix that is added to each address from a particular VPN site, thereby creating an address that is unique both within the VPN and within the Internet.

In addition, each VPN has its own VPN-specific routing table that contains the routing information for that VPN only. To separate VPN routes from routes in the Internet or those in other VPNs, the PE router creates a separate routing table for each VPN called a VPN routing and forwarding (VRF) table. The PE router creates one VRF table for each VPN that has a connection to a customer edge (CE) router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN. Every VRF table has one or more extended community attributes associated with it that identify the route as belonging to a specific collection of routers. One of these, the *route target* attribute, identifies a collection of sites (VRF tables) to which a PE router distributes routes. The PE router uses the route target to constrain the import of remote routes into its VRF tables.

When an ingress PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN.

- If the route matches, the route is converted to VPN-IPv4 format—that is, the route distinguisher is added to the route. The PE router then announces the route in VPN-IPv4 format to the remote PE routers. It also attaches a route target to each route learned from the directly connected sites. The route target attached to each route is based on the configured export target policy of the VRF table. The routes are then distributed using IBGP sessions, which are configured in the provider's core network.
- If the route from the CE router does not match, it is not exported to other PE routers, but it can still be used locally for routing, for example, if two CE routers in the same VPN are directly connected to the same PE router.

When an egress PE router receives a route, it checks it against the import policy on the IBGP session between the PE routers. If it is accepted, the router places the route into its `bgp.l3vpn.0` table. At the same time, the router checks the route against the VRF import policy for the VPN. If it matches, the route distinguisher is removed from the route and the route is placed into the VRF table (the *routing-instance-name.inet.0* table) in IPv4 format.

#### Related Documentation

- [Layer 2 VPN Overview](#)
- [Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview on page 468](#)

- [Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 468](#)
- [Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 493](#)
- [Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN on page 470](#)

---

## Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN

---

MPLS-based Layer 2 services are growing in demand among enterprise and service providers. This creates new challenges related to interoperability between Layer 2 and Layer 3 services for service providers who want to provide end-to-end value-added services. There are various reasons to stitch different Layer 2 services to one another and to Layer 3 services. For example, to expand the service offerings and to expand geographically. The Junos OS has various features to address the needs of the service provider.

Interconnecting a Layer 2 Circuit with a Layer 3 VPN provides the following benefits:

- Interconnecting a Layer 2 Circuit with a Layer 3 VPN enables the sharing of a service provider's core network infrastructure between IP and Layer 2 circuit services, reducing the cost of providing those services. A Layer 2 MPLS circuit allows service providers to create a Layer 2 circuit service over an existing IP and MPLS backbone.
- Service providers do not have to invest in separate Layer 2 equipment to provide Layer 2 circuit service. A service provider can configure a provider edge router to run any Layer 3 protocol in addition to the Layer 2 protocols. Customers who prefer to maintain control over most of the administration of their own networks want Layer 2 circuit connections with their service provider instead of a Layer 3 VPN connection.

### Related Documentation

- [Layer 3 VPN Overview on page 466](#)
- [Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 493](#)

---

## Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview

---

As MPLS-based Layer 2 services grow in demand, new challenges arise for service providers to be able to interoperate with Layer 2 and Layer 3 services and give their customers value-added services. Junos OS has various features to address the needs of service providers. One of these features is the use of a logical tunnel interface. This Junos OS functionality makes use of a tunnel PIC to loop packets out and back from the Packet Forwarding Engine to link the Layer 2 network with the Layer 3 network. The solution is limited by the logical tunnel bandwidth constraints imposed by the tunnel PIC.

## Interconnecting Layer 2 VPNs with Layer 3 VPNs Applications

Interconnecting a Layer 2 VPN with a Layer 3 VPN provides the following benefits:

- A single access line to provide multiple services—Traditional VPNs over Layer 2 circuits require the provisioning and maintenance of separate networks for IP and for VPN services. In contrast, Layer 2 VPNs enable the sharing of a provider's core network infrastructure between IP and Layer 2 VPN services, thereby reducing the cost of providing those services.
- Flexibility—Many different types of networks can be accommodated by the service provider. If all sites in a VPN are owned by the same enterprise, this is an intranet. If various sites are owned by different enterprises, the VPN is an extranet. A site can be located in more than one VPN.
- Wide range of possible policies—You can give every site in a VPN a different route to every other site, or you can force traffic between certain pairs of sites routed via a third site and so pass certain traffic through a firewall.
- Scalable network—This design enhances the scalability because it eliminates the need for provider edge (PE) routers to maintain all of the service provider's VPN routes. Each PE router maintains a VRF table for each of its directly connected sites. Each customer connection (such as a Frame Relay PVC, an ATM PVC, or a VLAN) is mapped to a specific VRF table. Thus, it is a port on the PE router and not a site that is associated with a VRF table. Multiple ports on a PE router can be associated with a single VRF table. It is the ability of PE routers to maintain multiple forwarding tables that supports the per-VPN segregation of routing information.
- Use of route reflectors—Provider edge routers can maintain IBGP sessions to route reflectors as an alternative to a full mesh of IBGP sessions. Deploying multiple route reflectors enhances the scalability of the RFC 2547bis model because it eliminates the need for any single network component to maintain all VPN routes.
- Multiple VPNs are kept separate and distinct from each other—The customer edge routers do not peer with each other. Two sites have IP connectivity over the common backbone only, and only if there is a VPN which contains both sites. This feature keeps the VPNs separate and distinct from each other, even if two VPNs have an overlapping address space.
- Simple for customers to use—Customers can obtain IP backbone services from a service provider, and they do not need to maintain their own backbones.

### Related Documentation

- [Layer 2 VPN Overview](#)
- [Layer 3 VPN Overview on page 466](#)
- [Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN on page 470](#)

## Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN

---

This example provides a step-by-step procedure and commands for interconnecting and verifying a Layer 2 VPN with a Layer 3 VPN. It contains the following sections:

- [Requirements on page 470](#)
- [Overview and Topology on page 470](#)
- [Configuration on page 473](#)
- [Verification on page 488](#)

### Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.3 or later
- Five MX Series routers
- Three M Series routers
- Two T Series routers

### Overview and Topology

A Layer 2 VPN is a type of virtual private network (VPN) that uses MPLS labels to transport data. The communication occurs between the provider edge (PE) routers.

Layer 2 VPNs use BGP as the signaling protocol and, consequently, have a simpler design and require less provisioning overhead than traditional VPNs over Layer 2 circuits. BGP signaling also enables autodiscovery of Layer 2 VPN peers. Layer 2 VPNs can have either a full-mesh or a hub-and-spoke topology. The tunneling mechanism in the core network is, typically, MPLS. However, Layer 2 VPNs can also use other tunneling protocols, such as GRE.

Layer 3 VPNs are based on RFC 2547bis, *BGP/MPLS IP VPNs*. RFC 2547bis defines a mechanism by which service providers can use their IP backbones to provide VPN services to their customers. A Layer 3 VPN is a set of sites that share common routing information and whose connectivity is controlled by a collection of policies. The sites that make up a Layer 3 VPN are connected over a provider's existing public Internet backbone. RFC 2547bis VPNs are also known as BGP/MPLS VPNs because BGP is used to distribute VPN routing information across the provider's backbone, and MPLS is used to forward VPN traffic across the backbone to remote VPN sites.

Customer networks, because they are private, can use either public addresses or private addresses, as defined in RFC 1918, *Address Allocation for Private Internets*. When customer networks that use private addresses connect to the public Internet infrastructure, the private addresses might overlap with the same private addresses used by other network users. MPLS/BGP VPNs solve this problem by adding a *route distinguisher*. A route distinguisher is a VPN identifier prefix that is added to each address from a particular VPN site, thereby creating an address that is unique both within the VPN and within the Internet.

In addition, each VPN has its own VPN-specific routing table that contains the routing information for that VPN only. To separate a VPN's routes from routes in the public Internet or those in other VPNs, the PE router creates a separate routing table for each VPN called a VPN routing and forwarding (VRF) table. The PE router creates one VRF table for each VPN that has a connection to a customer edge (CE) router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN. Every VRF table has one or more extended community attributes associated with it that identify the route as belonging to a specific collection of routers. One of these, the *route target* attribute, identifies a collection of sites (VRF tables) to which a PE router distributes routes. The PE router uses the route target to constrain the import of remote routes into its VRF tables.

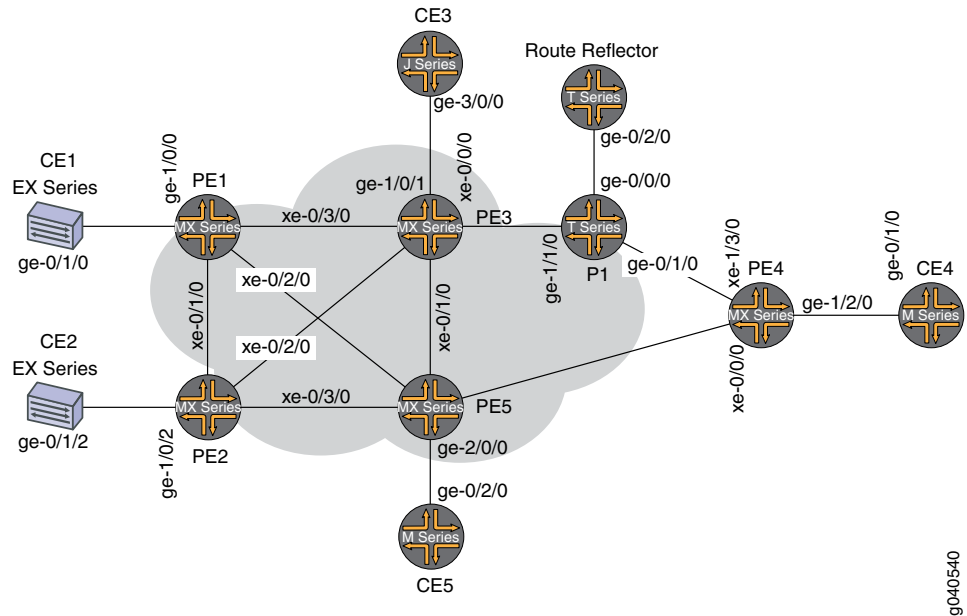
When an ingress PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN.

- If it matches, the route is converted to VPN-IPv4 format—that is, the route distinguisher is added to the route. The PE router then announces the route in VPN-IPv4 format to the remote PE routers. It also attaches a route target to each route learned from the directly connected sites. The route target attached to the route is based on the value of the VRF table's configured export target policy. The routes are then distributed using IBGP sessions, which are configured in the provider's core network.
- If the route from the CE router does not match, it is not exported to other PE routers, but it can still be used locally for routing, for example, if two CE routers in the same VPN are directly connected to the same PE router.

When an egress PE router receives a route, it checks it against the import policy on the IBGP session between the PE routers. If it is accepted, the router places the route into its `bgp.l3vpn.0` table. At the same time, the router checks the route against the VRF import policy for the VPN. If it matches, the route distinguisher is removed from the route and the route is placed into the VRF table (the *routing-instance-name.inet.0* table) in IPv4 format.

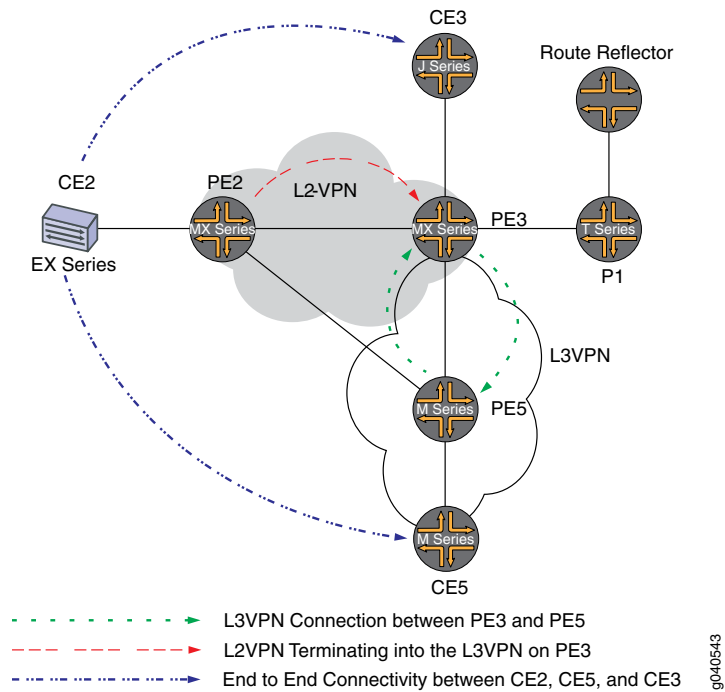
[Figure 58 on page 472](#) shows the physical topology of a Layer 2 VPN-to-Layer 3 VPN interconnection.

Figure 58: Physical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN



The logical topology of a Layer 2 VPN-to-Layer 3 VPN interconnection is shown in [Figure 59 on page 472](#).

Figure 59: Logical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN



The following definitions describe the meaning of the device abbreviations used in [Figure 58 on page 472](#) and [Figure 59 on page 472](#).

- Customer edge (CE) device—A device at the customer premises that provides access to the service provider's VPN over a data link to one or more provider edge (PE) routers.

Typically the CE device is an IP router that establishes an adjacency with its directly connected PE routers. After the adjacency is established, the CE router advertises the site's local VPN routes to the PE router and learns remote VPN routes from the PE router.

- Provider edge (PE) device—A device, or set of devices, at the edge of the provider network that presents the provider's view of the customer site.

PE routers exchange routing information with CE routers. PE routers are aware of the VPNs that connect through them, and PE routers maintain VPN state. A PE router is only required to maintain VPN routes for those VPNs to which it is directly attached. After learning local VPN routes from CE routers, a PE router exchanges VPN routing information with other PE routers using IBGP. Finally, when using MPLS to forward VPN data traffic across the provider's backbone, the ingress PE router functions as the ingress label-switching router (LSR) and the egress PE router functions as the egress LSR.

- Provider (P) device—A device that operates inside the provider's core network and does not directly interface to any CE.

Although the P device is a key part of implementing VPNs for the service provider's customers and may provide routing for many provider-operated tunnels that belong to different VPNs, it is not itself VPN-aware and does not maintain VPN state. Its principal role is allowing the service provider to scale its VPN offerings, for example, by acting as an aggregation point for multiple PE routers.

P routers function as MPLS transit LSRs when forwarding VPN data traffic between PE routers. P routers are required only to maintain routes to the provider's PE routers; they are not required to maintain specific VPN routing information for each customer site.

## Configuration

To interconnect a Layer 2 VPN with a Layer 3 VPN, perform these tasks:

- [Configuring the Base Protocols and Interfaces on page 473](#)
- [Configuring the VPN Interfaces on page 477](#)

### Configuring the Base Protocols and Interfaces

#### Step-by-Step Procedure

1. On each PE and P router, configure OSPF with traffic engineering extensions on all interfaces. Disable OSPF on the fxp0.0 interface.

```
[edit protocols]
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface all;
```

```
        interface fxp0.0 {  
            disable;  
        }  
    }  
}
```

2. On all the core routers, enable MPLS on all interfaces. Disable MPLS on the fxp0.0 interface.

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

3. On all the core routers, create an internal BGP peer group and specify the route reflector address (7.7.7.7) as the neighbor. Also enable BGP to carry Layer 2 VPLS network layer reachability information (NLRI) messages for this peer group by including the **signaling** statement at the **[edit protocols bgp group group-name family l2vpn]** hierarchy level.

```
[edit protocols]  
bgp {  
    group RR {  
        type internal;  
        local-address 2.2.2.2;  
        family l2vpn {  
            signaling;  
        }  
        neighbor 7.7.7.7;  
    }  
}
```

4. On Router PE3, create an internal BGP peer group and specify the route reflector IP address (7.7.7.7) as the neighbor. Enable BGP to carry Layer 2 VPLS NLRI messages for this peer group and enable the processing of VPN-IPv4 addresses by including the **unicast** statement at the **[edit protocols bgp group group-name family inet-vpn]** hierarchy level.

```
[edit protocols]  
bgp {  
    group RR {  
        type internal;  
        local-address 3.3.3.3;  
        family inet-vpn {  
            unicast;  
        }  
        family l2vpn {  
            signaling;  
        }  
        neighbor 7.7.7.7;  
    }  
}
```

5. For the Layer 3 VPN domain on Router PE3 and Router PE5, enable RSVP on all interfaces. Disable RSVP on the fxp0.0 interface.

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

6. On Router PE3 and Router PE5, create label-switched paths (LSPs) to the route reflector and the other PE routers. The following example shows the configuration on Router PE5.

```
[edit protocols]
mpls {
  label-switched-path to-RR {
    to 7.7.7.7;
  }
  label-switched-path to-PE2 {
    to 2.2.2.2;
  }
  label-switched-path to-PE3 {
    to 3.3.3.3;
  }
  label-switched-path to-PE4 {
    to 4.4.4.4;
  }
  label-switched-path to-PE1 {
    to 1.1.1.1;
  }
}
```

7. On Routers PE1, PE2, PE3, and PE5, configure the core interfaces with an IPv4 address and enable the MPLS address family. The following example shows the configuration of the xe-0/1/0 interface on Router PE2.

```
[edit]
interfaces {
  xe-0/1/0 {
    unit 0 {
      family inet {
        address 10.10.2.2/30;
      }
      family mpls;
    }
  }
}
```

8. On Router PE2 and Router PE3, configure LDP for the Layer 2 VPN MPLS signaling protocol for all interfaces. Disable LDP on the fxp0.0 interface. (RSVP can also be used.)

```
[edit protocols]
ldp {
  interface all;
```

```
interface fxp0.0 {  
  disable;  
}  
}
```

9. On the route reflector, create an internal BGP peer group and specify the PE routers IP addresses as the neighbors.

```
[edit]  
protocols {  
  bgp {  
    group RR {  
      type internal;  
      local-address 7.7.7.7;  
      family inet {  
        unicast;  
      }  
      family inet-vpn {  
        unicast;  
      }  
      family l2vpn {  
        signaling;  
      }  
      cluster 7.7.7.7;  
      neighbor 1.1.1.1;  
      neighbor 2.2.2.2;  
      neighbor 4.4.4.4;  
      neighbor 5.5.5.5;  
      neighbor 3.3.3.3;  
    }  
  }  
}
```

10. On the route reflector, configure MPLS LSPs towards Routers PE3 and PE5 to resolve the BGP next hops from inet.3 routing table.

```
[edit]  
protocols {  
  mpls {  
    label-switched-path to-pe3 {  
      to 3.3.3.3;  
    }  
    label-switched-path to-pe5 {  
      to 5.5.5.5;  
    }  
    interface all;  
  }  
}
```

### Configuring the VPN Interfaces

#### Step-by-Step Procedure

Router PE2 is one end of the Layer 2 VPN. Router PE3 is performing the Layer 2 VPN stitching between the Layer 2 VPN and the Layer 3 VPN. Router PE3 uses the logical tunnel interface (lt interface) configured with different logical interface units applied under two different Layer 2 VPN instances. The packet is looped though the lt interface configured on Router PE3. The configuration of Router PE5 contains the PE-CE interface.

1. On Router PE2, configure the ge-1/0/2 interface encapsulation. Include the encapsulation statement and specify the **ethernet-ccc** option (**vlan-ccc** encapsulation is also supported) at the **[edit interfaces ge-1/0/2]** hierarchy level. The encapsulation should be the same in a whole Layer 2 VPN domain (Routers PE2 and PE3). Also, configure interface lo0.

```
[edit]
interfaces {
  ge-1/0/2 {
    encapsulation ethernet-ccc;
    unit 0;
  }
  lo0 {
    unit 0 {
      family inet {
        address 2.2.2.2/32;
      }
    }
  }
}
```

2. On Router PE2, configure the routing instance at the **[edit routing-instances]** hierarchy level. Also, configure the Layer 2 VPN protocol at the **[edit routing-instances routing-instances-name protocols]** hierarchy level. Configure the remote site ID as 3. Site ID 3 represents Router PE3 (Hub-PE). The Layer 2 VPN is using LDP as the signaling protocol. Be aware that in the following example, both the routing instance and the protocol are named **l2vpn**.

```
[edit]
routing-instances {
  l2vpn { # routing instance
    instance-type l2vpn;
    interface ge-1/0/2.0;
    route-distinguisher 65000:2;
    vrf-target target:65000:2;
    protocols {
      l2vpn { # protocol
        encapsulation-type ethernet;
        site CE2 {
          site-identifier 2;
          interface ge-1/0/2.0 {
            remote-site-id 3;
          }
        }
      }
    }
  }
}
```

```
    }
  }
```

3. On Router PE5, configure the Gigabit Ethernet interface for the PE-CE link **ge-2/0/0** and configure the **lo0** interface.

```
[edit interfaces]
ge-2/0/0 {
  unit 0 {
    family inet {
      address 80.80.80.1/24;
    }
  }
}
lo0 {
  unit 0 {
  }
}
```

4. On Router PE5, configure the Layer 3 VPN routing instance (**L3VPN**) at the **[edit routing-instances]** hierarchy level. Also configure BGP at the **[edit routing-instances L3VPN protocols]** hierarchy level.

```
[edit]
routing-instances {
  L3VPN {
    instance-type vrf;
    interface ge-2/0/0.0;
    route-distinguisher 65000:5;
    vrf-target target:65000:2;
    vrf-table-label;
    protocols {
      bgp {
        group ce5 {
          neighbor 80.80.80.2 {
            peer-as 200;
          }
        }
      }
    }
  }
}
```

5. In an MX Series router, such as Router PE3, you must create the tunnel services interface to be used for tunnel services. To create the tunnel service interface, include the **bandwidth** statement and specify the amount of bandwidth to reserve for tunnel services in gigabits per second at the **[edit chassis fpc slot-number pic slot-number tunnel-services]** hierarchy level.

```
[edit]
chassis {
  dump-on-panic;
  fpc 1 {
    pic 1 {
      tunnel-services {
        bandwidth 1g;
      }
    }
  }
}
```

```

    }
  }
}

```

6. On Router PE3, configure the Gigabit Ethernet interface.

Include the **address** statement at the **[edit interfaces ge-1/0/1.0 family inet]** hierarchy level and specify **90.90.90.1/24** as the IP address.

```

[edit]
interfaces {
  ge-1/0/1 {
    unit 0 {
      family inet {
        address 90.90.90.1/24;
      }
    }
  }
}

```

7. On Router PE3, configure the **lt-1/1/10.0** logical tunnel interface at the **[edit interfaces lt-1/1/10 unit 0]** hierarchy level. Router PE3 is the router that is *stitching* the Layer 2 VPN to the Layer 3 VPN using the logical tunnel interface. The configuration of the peer unit interfaces is what makes the interconnection.

To configure the interface, include the **encapsulation** statement and specify the **ethernet-ccc** option. Include the **peer-unit** statement and specify the logical interface unit 1 as the peer tunnel interface. Include the **family** statement and specify the **ccc** option.

```

[edit]
interfaces {
  lt-1/1/10 {
    unit 0 {
      encapsulation ethernet-ccc;
      peer-unit 1;
      family ccc;
    }
  }
}

```

8. On Router PE3, configure the **lt-1/1/10.1** logical tunnel interface at the **[edit interfaces lt-1/1/10 unit 1]** hierarchy level.

To configure the interface, include the **encapsulation** statement and specify the **ethernet** option. Include the **peer-unit** statement and specify the logical interface unit 0 as the peer tunnel interface. Include the **family** statement and specify the **inet** option. Include the **address** statement at the **[edit interfaces lt-1/1/10 unit 0]** hierarchy level and specify **70.70.70.1/24** as the IPv4 address.

```

[edit]
interfaces {
  lt-1/1/10 {
    unit 1 {
      encapsulation ethernet;
      peer-unit 0;
      family inet {

```

```

        address 70.70.70.1/24;
    }
}
}
}

```

9. On Router PE3, add the **lt** interface unit 1 to the routing instance at the **[edit routing-instances L3VPN]** hierarchy level. Configure the instance type as **vrf** with **lt** peer-unit 1 as a PE-CE interface to terminate the Layer 2 VPN on Router PE2 into the Layer 3 VPN on Router PE3.

```

[edit]
routing-instances {
  L3VPN {
    instance-type vrf;
    interface ge-1/0/1.0;
    interface lt-1/1/10.1;
    route-distinguisher 65000:33;
    vrf-target target:65000:2;
    vrf-table-label;
    protocols {
      bgp {
        export direct;
        group ce3 {
          neighbor 90.90.90.2 {
            peer-as 100;
          }
        }
      }
    }
  }
}
}

```

10. On Router PE3, add the **lt** interface unit 0 to the routing instance at the **[edit routing-instances protocols l2vpn]** hierarchy level. Also configure the same vrf target for the Layer 2 VPN and Layer 3 VPN routing instances, so that the routes can be leaked between the instances. The example configuration in the previous step shows the vrf target for the **L3VPN** routing instance. The following example shows the vrf target for the **l2vpn** routing instance.

```

[edit]
routing-instances {
  l2vpn {
    instance-type l2vpn;
    interface lt-1/1/10.0;
    route-distinguisher 65000:3;
    vrf-target target:65000:2;
    protocols {
      l2vpn {
        encapsulation-type ethernet;
        site CE3 {
          site-identifier 3;
          interface lt-1/1/10.0 {
            remote-site-id 2;
          }
        }
      }
    }
  }
}

```

```

    }
  }
}

```

11. On Router PE3, configure the **policy-statement** statement to export the routes learned from the directly connected **lt** interface unit 1 to all the CE routers for connectivity, if needed.

```

[edit]
policy-options {
  policy-statement direct {
    term 1 {
      from protocol direct;
      then accept;
    }
  }
}

```

**Results** The following output shows the full configuration of Router PE2:

```

Router PE2 interfaces {
  xe-0/1/0 {
    unit 0 {
      family inet {
        address 10.10.2.2/30;
      }
      family mpls;
    }
  }
  xe-0/2/0 {
    unit 0 {
      family inet {
        address 10.10.5.1/30;
      }
      family mpls;
    }
  }
  xe-0/3/0 {
    unit 0 {
      family inet {
        address 10.10.4.1/30;
      }
      family mpls;
    }
  }
  ge-1/0/2 {
    encapsulation ethernet-ccc;
    unit 0;
  }
  fxp0 {
    apply-groups [ re0 re1 ];
  }
  lo0 {
    unit 0 {
      family inet {

```

```
        address 2.2.2.2/32;
    }
}
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group RR {
            type internal;
            local-address 2.2.2.2;
            family l2vpn {
                signaling;
            }
            neighbor 7.7.7.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
routing-instances {
    l2vpn {
        instance-type l2vpn;
        interface ge-1/0/2.0;
        route-distinguisher 65000:2;
        vrf-target target:65000:2;
        protocols {
            l2vpn {
                encapsulation-type ethernet;
            }
            site CE2 {
                site-identifier 2;
                interface ge-1/0/2.0 {
```

```

        remote-site-id 3;
    }
}
}
}
}
}

```

The following output shows the final configuration of Router PE5:

```

Router PE5  interfaces {
              ge-0/0/0 {
                unit 0 {
                  family inet {
                    address 10.10.4.2/30;
                  }
                  family mpls;
                }
              }
              xe-0/1/0 {
                unit 0 {
                  family inet {
                    address 10.10.6.2/30;
                  }
                  family mpls;
                }
              }
              ge-1/0/0 {
                unit 0 {
                  family inet {
                    address 10.10.9.1/30;
                  }
                  family mpls;
                }
              }
              xe-1/1/0 {
                unit 0 {
                  family inet {
                    address 10.10.3.2/30;
                  }
                  family mpls;
                }
              }
              ge-2/0/0 {
                unit 0 {
                  family inet {
                    address 80.80.80.1/24;
                  }
                }
              }
              lo0 {
                unit 0 {
                  family inet {
                    address 5.5.5.5/32;
                  }
                }
              }
            }

```

```
    }  
  }  
  routing-options {  
    static {  
      route 172.0.0.0/8 next-hop 172.19.59.1;  
    }  
    autonomous-system 65000;  
  }  
  protocols {  
    rsvp {  
      interface all {  
        link-protection;  
      }  
      interface fxp0.0 {  
        disable;  
      }  
    }  
    mpls {  
      label-switched-path to-RR {  
        to 7.7.7.7;  
      }  
      label-switched-path to-PE2 {  
        to 2.2.2.2;  
      }  
      label-switched-path to-PE3 {  
        to 3.3.3.3;  
      }  
      label-switched-path to-PE4 {  
        to 4.4.4.4;  
      }  
      label-switched-path to-PE1 {  
        to 1.1.1.1;  
      }  
      interface all;  
      interface fxp0.0 {  
        disable;  
      }  
    }  
    bgp {  
      group to-rr {  
        type internal;  
        local-address 5.5.5.5;  
        family inet-vpn {  
          unicast;  
        }  
        family l2vpn {  
          signaling;  
        }  
        neighbor 7.7.7.7;  
      }  
    }  
    ospf {  
      traffic-engineering;  
      area 0.0.0.0 {  
        interface all;  
        interface fxp0.0 {
```

```

        disable;
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-2/0/0.0;
        route-distinguisher 65000:5;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                group ce5 {
                    neighbor 80.80.80.2 {
                        peer-as 200;
                    }
                }
            }
        }
    }
}
}

```

The following output shows the final configuration of Router PE3:

```

Router PE3  chassis {
              dump-on-panic;
              fpc 1 {
                  pic 1 {
                      tunnel-services {
                          bandwidth 1g;
                      }
                  }
              }
              network-services ip;
          }
          interfaces {
              ge-1/0/1 {
                  unit 0 {
                      family inet {
                          address 90.90.90.1/24;
                      }
                  }
              }
              lt-1/1/10 {
                  unit 0 {
                      encapsulation ethernet-ccc;
                      peer-unit 1;
                      family ccc;
                  }
              }
          }
      }
  }

```

```
}
unit 1 {
    encapsulation ethernet;
    peer-unit 0;
    family inet {
        address 70.70.70.1/24;
    }
}
}
xe-2/0/0 {
    unit 0 {
        family inet {
            address 10.10.20.2/30;
        }
        family mpls;
    }
}
xe-2/1/0 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
        }
        family mpls;
    }
}
xe-2/2/0 {
    unit 0 {
        family inet {
            address 10.10.5.2/30;
        }
        family mpls;
    }
}
xe-2/3/0 {
    unit 0 {
        family inet {
            address 10.10.1.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 3.3.3.3/32;
        }
    }
}
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
```

```
    rsvp {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
    mpls {
      label-switched-path to-RR {
        to 7.7.7.7;
      }
      label-switched-path to-PE2 {
        to 2.2.2.2;
      }
      label-switched-path to-PE5 {
        to 5.5.5.5;
      }
      label-switched-path to-PE4 {
        to 4.4.4.4;
      }
      label-switched-path to-PE1 {
        to 1.1.1.1;
      }
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
    bgp {
      group RR {
        type internal;
        local-address 3.3.3.3;
        family inet-vpn {
          unicast;
        }
        family l2vpn {
          signaling;
        }
        neighbor 7.7.7.7;
      }
    }
    ospf {
      traffic-engineering;
      area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
          disable;
        }
      }
    }
    ldp {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
  }
}
```

```

policy-options {
  policy-statement direct {
    term 1 {
      from protocol direct;
      then accept;
    }
  }
}
routing-instances {
  L3VPN {
    instance-type vrf;
    interface ge-1/0/1.0;
    interface lt-1/1/10.1;
    route-distinguisher 65000:33;
    vrf-target target:65000:2;
    vrf-table-label;
    protocols {
      bgp {
        export direct;
        group ce3 {
          neighbor 90.90.90.2 {
            peer-as 100;
          }
        }
      }
    }
  }
  l2vpn {
    instance-type l2vpn;
    interface lt-1/1/10.0;
    route-distinguisher 65000:3;
    vrf-target target:65000:2;
    protocols {
      l2vpn {
        encapsulation-type ethernet;
        site CE3 {
          site-identifier 3;
          interface lt-1/1/10.0 {
            remote-site-id 2;
          }
        }
      }
    }
  }
}

```

## Verification

Verify the Layer 2 VPN-to-Layer 3 VPN interconnection:

- [Verifying Router PE2 VPN Interface on page 489](#)
- [Verifying Router PE3 VPN Interface on page 490](#)
- [Verifying End-to-End connectivity from Router CE2 to Router CE5 and Router CE3 on page 492](#)

### Verifying Router PE2 VPN Interface

**Purpose** Check that the Layer 2 VPN is up and working at the Router PE2 interface and that all the routes are there.

- Action** 1. Use the **show l2vpn connections** command to verify that the connection site ID is 3 for Router PE3 and that the status is **Up**.

```
user@PE2> show l2vpn connections
Layer-2 VPN connections:
Legend for connection status (St)
EI -- encapsulation invalid      NC -- interface encapsulation not CCC/TCC/VPLS
EM -- encapsulation mismatch     WE -- interface and instance encaps not same
VC-Dn -- Virtual circuit down   NP -- interface hardware not present
CM -- control-word mismatch     -> -- only outbound connection is up
CN -- circuit not provisioned   <- -- only inbound connection is up
OR -- out of range             Up -- operational
OL -- no outgoing label        Dn -- down
LD -- local site signaled down  CF -- call admission control failure
RD -- remote site signaled down SC -- local and remote site ID collision
LN -- local site not designated LM -- local site ID not minimum designated
RN -- remote site not designated RM -- remote site ID not minimum designated
XX -- unknown connection status IL -- no incoming label
MM -- MTU mismatch            MI -- Mesh-Group ID not available
BK -- Backup connection        ST -- Standby connection
PF -- Profile parse failure    PB -- Profile busy
RS -- remote site standby
```

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

```
Instance: l2vpn
Local site: CE2 (2)
  connection-site  Type  St    Time last up      # Up trans
    3              rmt   Up    Jan 7 14:14:37 2010      1
  Remote PE: 3.3.3.3, Negotiated control-word: Yes (Null)
  Incoming label: 800000, Outgoing label: 800001
  Local interface: ge-1/0/2.0, Status: Up, Encapsulation: ETHERNET
```

2. Use the **show route table** command to verify that the Layer 2 VPN route is present and that there is a next hop of **10.10.5.2** through the **xe-0/2/0.0** interface. The following output verifies that the Layer 2 VPN routes are present in the **l2vpn.l2vpn.0** table. Similar output should be displayed for Router PE3.

```
user@PE2> show route table l2vpn.l2vpn.0
l2vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

65000:2:2:3/96
    *[L2VPN/170/-101] 02:40:35, metric2 1
    Indirect
65000:3:3:1/96
    *[BGP/170] 02:40:35, localpref 100, from 7.7.7.7
    AS path: I
    > to 10.10.5.2 via xe-0/2/0.0
```

3. Verify that Router PE2 has a Layer 2 VPN MPLS label pointing to the LDP label to Router PE3 in both directions (PUSH and POP).

```

user@PE2> show route table mpls.0
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0          *[MPLS/0] 1w3d 08:57:41, metric 1
           Receive
1          *[MPLS/0] 1w3d 08:57:41, metric 1
           Receive
2          *[MPLS/0] 1w3d 08:57:41, metric 1
           Receive
300560     *[LDP/9] 19:45:53, metric 1
           > to 10.10.2.1 via xe-0/1/0.0, Pop
300560(S=0) *[LDP/9] 19:45:53, metric 1
           > to 10.10.2.1 via xe-0/1/0.0, Pop
301008     *[LDP/9] 19:45:53, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Swap 299856
301536     *[LDP/9] 19:45:53, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Pop
301536(S=0) *[LDP/9] 19:45:53, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Pop
301712     *[LDP/9] 16:14:52, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Swap 315184
301728     *[LDP/9] 16:14:52, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Pop
301728(S=0) *[LDP/9] 16:14:52, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Pop
800000     *[L2VPN/7] 02:40:35
           > via ge-1/0/2.0, Pop Offset: 4
ge-1/0/2.0 *[L2VPN/7] 02:40:35, metric 2 1
           > to 10.10.5.2 via xe-0/2/0.0, Push 800001 Offset: -4

```

**Meaning** The `l2vpn` routing instance is up at interface `ge-1/0/2` and the Layer 2 VPN route is shown in table `l2vpn.l2vpn.0`. Table `mpls.0` shows the Layer 2 VPN routes used to forward the traffic using an LDP label.

### Verifying Router PE3 VPN Interface

**Purpose** Check that the Layer 2 VPN connection from Router PE2 and Router PE3 is **Up** and working.

**Action** 1. Verify that the BGP session with the route reflector for the family `l2vpn-signaling` and the family `inet-vpn` is established.

```

user@PE3> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table      Tot Paths  Act Paths Suppressed  History Damp State   Pending
bgp.l2vpn.0      1          1          0          0          0          0
bgp.L3VPN.0       1          1          0          0          0          0
Peer      AS    InPkt   OutPkt   OutQ   Flaps  Last Up/Dwn  State|#Active /Received/Accepted/Damped...
7.7.7.7  65000   2063    2084     0       1    15:35:16   Establ
  bgp.l2vpn.0: 1/1/1/0
  bgp.L3VPN.0: 1/1/1/0
  L3VPN.inet.0: 1/1/1/0
  l2vpn.l2vpn.0: 1/1/1/0

```

2. The following output shows the `L3VPN.inet.0` routing table, which has Routers CE1, CE3, and CE5 listed.

```

user@PE3> show route table L3VPN.inet.0
L3VPN.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

70.70.70.0/24      *[Direct/0] 02:45:16
                  > via lt-1/1/10.1
70.70.70.1/32     *[Local/0] 14:45:42
                  Local via lt-1/1/10.1
80.80.80.0/24     *[BGP/170] 02:47:51, localpref 100, from 7.7.7.7
                  AS path: I
                  > to 10.10.6.2 via xe-2/1/0.0, Push 16
90.90.90.0/24     *[Direct/0] 15:26:24
                  > via ge-1/0/1.0
90.90.90.1/32     *[Local/0] 15:26:24
                  Local via ge-1/0/1.0

```

3. The following output verifies the Layer 2 VPN route and the label associated with it.

```

user@PE3> show route table l2vpn.l2vpn.0 detail
l2vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
65000:2:2:3/96 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Route Distinguisher: 65000:2
            Next hop type: Indirect
            Next-hop reference count: 4
            Source: 7.7.7.7
            Protocol next hop: 2.2.2.2
            Indirect next hop: 2 no-forward
            State: <Secondary Active Int Ext>
            Local AS: 65000 Peer AS: 65000
            Age: 2:45:52 Metric2: 1
            Task: BGP_65000.7.7.7+60585
            Announcement bits (1): 0-l2vpn-l2vpn
            AS path: I (Originator) Cluster list: 7.7.7.7
            AS path: Originator ID: 2.2.2.2
            Communities: target:65000:2 Layer2-info: encaps:ETHERNET,
control flags:Control-Word, mtu: 0, site preference: 100 Accepted
            Label-base: 800000, range: 2, status-vector: 0x0
            Localpref: 100
            Router ID: 7.7.7.7
            Primary Routing Table bgp.l2vpn.0

```

4. The following output show the L2VPN MPLS.0 route in the mpls.0 route table.

```

user@PE3> show route table mpls.0
mpls.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0              *[MPLS/0] 1w3d 09:05:41, metric 1
              Receive
1              *[MPLS/0] 1w3d 09:05:41, metric 1
              Receive
2              *[MPLS/0] 1w3d 09:05:41, metric 1
              Receive
16             *[VPN/0] 15:59:24
              to table L3VPN.inet.0, Pop
315184         *[LDP/9] 16:21:53, metric 1
              > to 10.10.20.1 via xe-2/0/0.0, Pop
315184(S=0)    *[LDP/9] 16:21:53, metric 1
              > to 10.10.20.1 via xe-2/0/0.0, Pop
315200         *[LDP/9] 01:13:44, metric 1
              to 10.10.20.1 via xe-2/0/0.0, Swap 625297

```

```

315216          > to 10.10.6.2 via xe-2/1/0.0, Swap 299856
                *[LDP/9] 16:21:53, metric 1
315216(S=0)     > to 10.10.6.2 via xe-2/1/0.0, Pop
                *[LDP/9] 16:21:53, metric 1
315232          > to 10.10.6.2 via xe-2/1/0.0, Pop
                *[LDP/9] 16:21:45, metric 1
315232(S=0)     > to 10.10.1.1 via xe-2/3/0.0, Pop
                *[LDP/9] 16:21:45, metric 1
315248          > to 10.10.1.1 via xe-2/3/0.0, Pop
                *[LDP/9] 16:21:53, metric 1
315248(S=0)     > to 10.10.5.1 via xe-2/2/0.0, Pop
                *[LDP/9] 16:21:53, metric 1
315312          > to 10.10.5.1 via xe-2/2/0.0, Pop
                *[LDP/9] 16:21:53, metric 1
315312(S=0)     > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
                *[RSVP/7] 15:02:40, metric 1
315328          > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
                *[RSVP/7] 15:02:40, metric 1
315360          > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
                *[RSVP/7] 15:02:40, metric 1
316272          > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
                *[RSVP/7] 01:13:27, metric 1
                > to 10.10.6.2 via xe-2/1/0.0, label-switched-path
Bypass->10.10.9.1
316272(S=0)     *[RSVP/7] 01:13:27, metric 1
                > to 10.10.6.2 via xe-2/1/0.0, label-switched-path
Bypass->10.10.9.1
800001          *[L2VPN/7] 02:47:33
                > via lt-1/1/10.0, Pop          Offset: 4
lt-1/1/10.0     *[L2VPN/7] 02:47:33, metric2 1
                > to 10.10.5.1 via xe-2/2/0.0, Push 800000 Offset: -4

```

5. Use the **show route table mpls.0** command with the **detail** option to see the BGP attributes of the route such as next-hop type and label operations.

```

user@PE5> show route table mpls.0 detail
lt-1/1/10.0 (1 entry, 1 announced)
    *L2VPN Preference: 7
        Next hop type: Indirect
        Next-hop reference count: 2
        Next hop type: Router, Next hop index: 607
        Next hop: 10.10.5.1 via xe-2/2/0.0, selected
        Label operation: Push 800000 Offset: -4
        Protocol next hop: 2.2.2.2
        Push 800000 Offset: -4
        Indirect next hop: 8cae0a0 1048574
        State: <Active Int>
        Age: 2:46:34 Metric2: 1
        Task: Common L2 VC
        Announcement bits (2): 0-KRT 2-Common L2 VC
        AS path: I
        Communities: target:65000:2 Layer2-info: encaps:ETHERNET,
control flags:Control-Word, mtu: 0, site preference: 100

```

### Verifying End-to-End connectivity from Router CE2 to Router CE5 and Router CE3

**Purpose** Check the connectivity between Routers CE2, CE3, and CE5.

**Action** 1. Ping the Router CE3 IP address from Router CE2.

```

user@CE2> ping 90.90.90.2 # CE3 IP address
PING 90.90.90.2 (90.90.90.2): 56 data bytes
64 bytes from 90.90.90.2: icmp_seq=0 ttl=63 time=0.708 ms
64 bytes from 90.90.90.2: icmp_seq=1 ttl=63 time=0.610 ms

```

2. Ping the Router CE5 IP address from Router CE2.

```

user@CE2> ping 80.80.80.2 # CE5 IP address
PING 80.80.80.2 (80.80.80.2): 56 data bytes
64 bytes from 80.80.80.2: icmp_seq=0 ttl=62 time=0.995 ms
64 bytes from 80.80.80.2: icmp_seq=1 ttl=62 time=1.005 ms

```

#### Related Documentation

- [Layer 2 VPN Overview](#)
- [Layer 3 VPN Overview on page 466](#)
- [Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview on page 468](#)

## Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN

This example provides a step-by-step procedure and commands for configuring and verifying a Layer 2 circuit to Layer 3 VPN interconnection. It contains the following sections:

- [Requirements on page 493](#)
- [Overview and Topology on page 493](#)
- [Configuration on page 495](#)
- [Verifying the Layer 2 Circuit to Layer 3 VPN Interconnection on page 505](#)

### Requirements

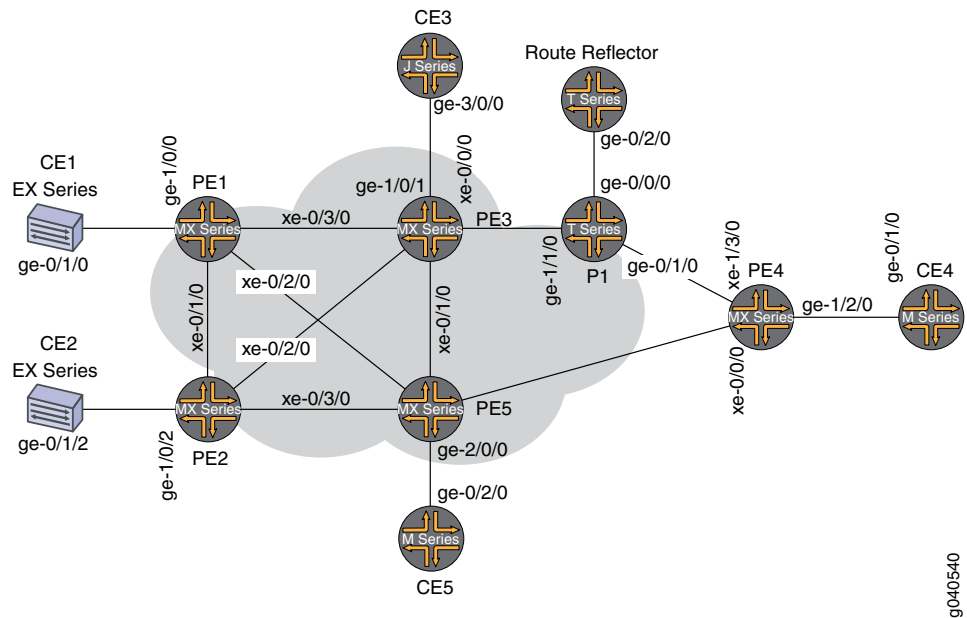
This example uses the following hardware and software components:

- Junos OS Release 9.3 or later
- 3 MX Series 3D Universal Edge Routers
- 1 M Series Multiservice Edge Router
- 1 T Series Core Router
- 1 EX Series Ethernet Switch
- 1 J Series Services Routers

### Overview and Topology

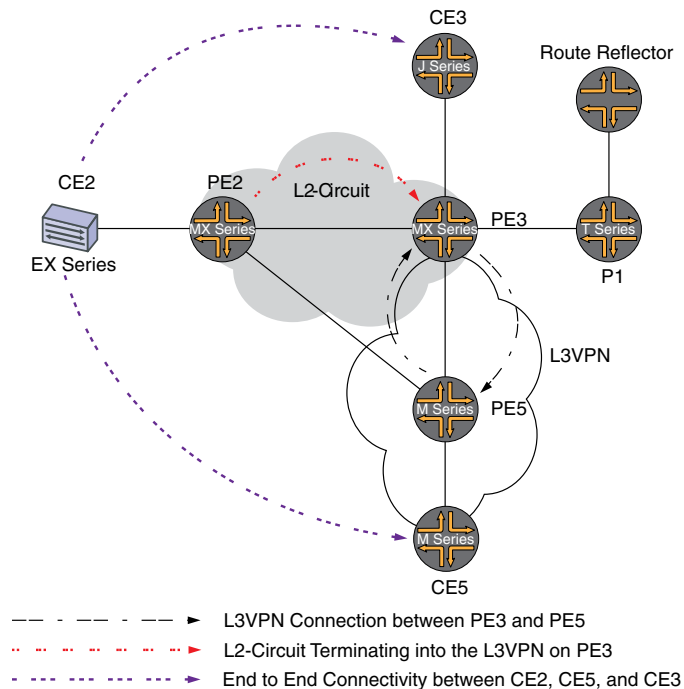
The physical topology of a Layer 2 circuit to Layer 3 VPN interconnection is shown in [Figure 60 on page 494](#).

Figure 60: Physical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection



The logical topology of a Layer 2 circuit to Layer 3 VPN interconnection is shown in [Figure 61 on page 494](#).

Figure 61: Logical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection



## Configuration



**NOTE:** In any configuration session, it is good practice to verify periodically that the configuration can be committed using the `commit check` command.

In this example, the router being configured is identified using the following command prompts:

- **CE2** identifies the customer edge 2 (CE2) router
- **PE1** identifies the provider edge 1 (PE1) router
- **CE3** identifies the customer edge 3 (CE3) router
- **PE3** identifies the provider edge 3 (PE3) router
- **CE5** identifies the customer edge 5 (CE5) router
- **PE5** identifies the provider edge 5 (PE5) router

This example contains the following procedures:

- [Configuring PE Router Customer-facing and Loopback Interfaces on page 495](#)
- [Configuring Core-facing Interfaces on page 497](#)
- [Configuring Protocols on page 498](#)
- [Configuring Routing Instances and Layer 2 Circuits on page 501](#)
- [Configuring the Route Reflector on page 503](#)
- [Interconnecting the Layer 2 Circuit with the Layer 3 VPN on page 504](#)

### Configuring PE Router Customer-facing and Loopback Interfaces

#### Step-by-Step Procedure

To begin building the interconnection, configure the interfaces on the PE routers. If your network contains provider (P) routers, configure the interfaces on the P routers also. This example shows the configuration for Router PE2, Router PE3, and Router PE5.

1. On Router PE2, configure the **ge-1/0/2** interface encapsulation. To configure the interface encapsulation, include the **encapsulation** statement and specify the **ethernet-ccc** option (**vlan-ccc** encapsulation is also supported). Configure the **ge-1/0/2.0** logical interface family for circuit cross-connect functionality. To configure the logical interface family, include the **family** statement and specify the **ccc** option. The encapsulation should be configured the same way for all routers in the Layer 2 circuit domain.

```
[edit interfaces]
ge-1/0/2 {
  encapsulation ethernet-ccc;
  unit 0 {
    family ccc;
  }
}
```

2. On Router PE2, configure the **lo0.0** interface. Include the **family** statement and specify the **inet** option. Include the **address** statement and specify **2.2.2.2/32** as the loopback IPv4 address.

```
[edit interfaces]
lo0 {
  unit 0 {
    family inet {
      address 2.2.2.2/32;
    }
  }
}
```

3. On Router PE3, configure the **ge-1/0/1** interface. Include the **family** statement and specify the **inet** option. Include the **address** statement and specify **90.90.90.1/24** as the interface address for this device.

```
[edit interfaces]
ge-1/0/1 {
  unit 0 {
    family inet {
      address 90.90.90.1/24;
    }
  }
}
```

4. On Router PE3, configure the **lo0.0** loopback interface. Include the **family** statement and specify the **inet** option. Include the **address** statement and specify **3.3.3.3/32** as the loopback IPv4 address for this router.

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

5. On Router PE5, configure the **ge-2/0/0** interface. Include the **family** statement and specify the **inet** option. Include the **address** statement and specify **80.80.80.1/24** as the interface address.

```
[edit interfaces]
ge-2/0/0 {
  unit 0 {
    family inet {
      address 80.80.80.1/24;
    }
  }
}
```

6. On Router PE5, configure the **lo0.0** interface. Include the **family** statement and specify the **inet** option. Include the **address** statement and specify **5.5.5.5/32** as the loopback IPv4 address for this router.

```
[edit interfaces]
lo0 {
```

```

    unit 0 {
        family inet {
            address 5.5.5.5/32;
        }
    }
}

```

### Configuring Core-facing Interfaces

#### Step-by-Step Procedure

This procedure describes how to configure the core-facing interfaces on the PE routers. This example does not include all the core-facing interfaces shown in the physical topology illustration. Enable the **mpls** and **inet** address families on the core-facing interfaces.

1. On Router PE2, configure the **xe-0/2/0** interface. Include the **family** statement and specify the **inet** address family. Include the **address** statement and specify **10.10.5.1/30** as the interface address. Include the **family** statement and specify the **mpls** address family.

```

[edit interfaces]
xe-0/2/0 {
    unit 0 {
        family inet {
            address 10.10.5.1/30;
        }
        family mpls;
    }
}

```

2. On Router PE3, configure the core-facing interfaces. Include the **family** statement and specify the **inet** address family. Include the **address** statement and specify the IPv4 addresses shown in the example as the interface addresses. Include the **family** statement and specify the **mpls** address family. In the example, the **xe-2/1/0** interface is connected to Router PE5, and the **xe-2/2/0** interface is connected to Router PE2.

```

[edit interfaces]
xe-2/0/0 {
    unit 0 {
        family inet {
            address 10.10.20.2/30;
        }
        family mpls;
    }
}
xe-2/1/0 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
        }
        family mpls;
    }
}
xe-2/2/0 {
    unit 0 {
        family inet {

```

```

        address 10.10.5.2/30;
    }
    family mpls;
}
xe-2/3/0 {
    unit 0 {
        family inet {
            address 10.10.1.2/30;
        }
        family mpls;
    }
}

```

3. On Router PE5, configure the **xe-0/1/0** interface. Include the **family** statement and specify the **inet** address family. Include the **address** statement and specify **10.10.6.2/30** as the interface address. Include the **family** statement and specify the **mpls** address family.

```

[edit interfaces]
xe-0/1/0 {
    unit 0 {
        family inet {
            address 10.10.6.2/30;
        }
        family mpls;
    }
}

```

### Configuring Protocols

#### Step-by-Step Procedure

This procedure describes how to configure the protocols used in this example. If your network contains P routers, configure the interfaces on the P routers also.

1. On Router PE3, enable OSPF as the IGP. Enable the MPLS, LDP, and BGP protocols on all interfaces except **fxp0.0**. LDP is used as the signaling protocol for the Layer 2 circuit to Router PE2. The following configuration snippet shows the protocol configuration for Router PE3:

```

[edit]
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 7.7.7.7;
        }
        label-switched-path to-PE2 {
            to 2.2.2.2;
        }
        label-switched-path to-PE5 {

```

```

        to 5.5.5.5;
    }
    label-switched-path to-PE4 {
        to 4.4.4.4;
    }
    label-switched-path to-PE1 {
        to 1.1.1.1;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group RR {
        type internal;
        local-address 3.3.3.3;
        family inet-vpn {
            unicast;
        }
        family l2vpn {
            signaling;
        }
        neighbor 7.7.7.7;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}

```

2. On Router PE2, configure the MPLS, OSPF, and LDP protocols.

```

[edit ]
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {

```

```

        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
}

```

3. On Router PE5, enable OSPF as the IGP. Enable the MPLS, RSVP, and BGP protocols on all interfaces except **fxp0.0**. Enable core-facing interfaces with the **mpls** and **inet** address families.

```

[edit]
protocols {
    rsvp {
        interface all {
            link-protection;
        }
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 7.7.7.7;
        }
        label-switched-path to-PE2 {
            to 2.2.2.2;
        }
        label-switched-path to-PE3 {
            to 3.3.3.3;
        }
        label-switched-path to-PE4 {
            to 4.4.4.4;
        }
        label-switched-path to-PE1 {
            to 1.1.1.1;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
bgp {
    group to-rr {
        type internal;
        local-address 5.5.5.5;
        family inet-vpn {
            unicast;
        }
    }
}

```

```

        family l2vpn {
            signaling;
        }
        neighbor 7.7.7.7;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
}
}

```

### Configuring Routing Instances and Layer 2 Circuits

#### Step-by-Step Procedure

This procedure describes how to configure the Layer 2 circuit and the Layer 3 VPN.

1. On Router PE2, configure the Layer 2 circuit. Include the **l2circuit** statement. Include the **neighbor** statement and specify the loopback IPv4 address of Router PE3 as the neighbor. Include the interface statement and specify **ge-1/0/2.0** as the logical interface that is participating in the Layer 2 circuit. Include the **virtual-circuit-id** statement and specify **100** as the identifier. Include the **no-control-word** statement for equipment that does not support the control word.

```

[edit ]
protocols {
    l2circuit {
        neighbor 3.3.3.3 {
            interface ge-1/0/2.0 {
                virtual-circuit-id 100;
                no-control-word;
            }
        }
    }
}
}

```

2. On Router PE3, configure the Layer 2 circuit to Router PE2. Include the **l2circuit** statement. Include the **neighbor** statement and specify the loopback IPv4 address of Router PE2 as the neighbor. Include the interface statement and specify **lt-1/1/10.0** as the logical tunnel interface that is participating in the Layer 2 circuit. Include the **virtual-circuit-id** statement and specify **100** as the identifier. Include the **no-control-word** statement.

```

[edit ]
protocols {
    l2circuit {
        neighbor 2.2.2.2 {
            interface lt-1/1/10.0 {
                virtual-circuit-id 100;
                no-control-word;
            }
        }
    }
}

```

```

    }
  }
}

```

3. On Router PE3, configure the Layer 3 VPN (**L3VPN**) routing instance to Router PE5 at the **[edit routing-instances]** hierarchy level. Also configure the BGP peer group at the **[edit routing-instances L3VPN protocols]** hierarchy level.

```

[edit ]
routing-instances {
  L3VPN {
    instance-type vrf;
    interface ge-1/0/1.0;
    interface lt-1/1/10.1;
    route-distinguisher 65000:33;
    vrf-target target:65000:2;
    vrf-table-label;
    protocols {
      bgp {
        export direct;
        group ce3 {
          neighbor 90.90.90.2 {
            peer-as 100;
          }
        }
      }
    }
  }
}

```

4. On Router PE5, configure the Layer 3 VPN routing instance (**L3VPN**) at the **[edit routing-instances]** hierarchy level. Also configure the BGP peer group at the **[edit routing-instances L3VPN protocols]** hierarchy level.

```

[edit ]
routing-instances {
  L3VPN {
    instance-type vrf;
    interface ge-2/0/0.0;
    route-distinguisher 65000:5;
    vrf-target target:65000:2;
    vrf-table-label;
    protocols {
      bgp {
        group ce5 {
          neighbor 80.80.80.2 {
            peer-as 200;
          }
        }
      }
    }
  }
}

```

### Configuring the Route Reflector

**Step-by-Step Procedure** Although a route reflector is not required to interconnect a Layer 2 circuit with a Layer 3 VPN, this examples uses a route reflector. This procedure shows the relevant portion of the route reflector configuration.

1. Configure the route reflector with RSVP, MPLS, BGP and OSPF. The route reflector is a BGP peer with the PE routers. Notice that the BGP peer group configuration includes the **family** statement and specifies the **inet-vpn** option. The **inet-vpn** option enables BGP to advertise network layer reachability information (NLRI) for the Layer 3 VPN routes. The configuration also includes the **family** statement and specifies the **l2vpn** option. The **l2vpn** option enables BGP to advertise NLRI for the Layer 2 circuit. Layer 2 circuits use the same internal BGP infrastructure as Layer 2 VPNs.

```
[edit ]
protocols {
  rsvp {
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  mpls {
    label-switched-path to-pe3 {
      to 3.3.3.3;
    }
    label-switched-path to-pe5 {
      to 5.5.5.5;
    }
    interface all;
    interface fxp0.0 {
      disable;
    }
  }
  bgp {
    group RR {
      type internal;
      local-address 7.7.7.7;
      family inet {
        unicast;
      }
      family inet-vpn {
        unicast;
      }
      family l2vpn {
        signaling;
      }
      cluster 7.7.7.7;
      neighbor 1.1.1.1;
      neighbor 2.2.2.2;
      neighbor 4.4.4.4;
      neighbor 5.5.5.5;
      neighbor 3.3.3.3;
    }
  }
}
```

```

    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
}

```

### Interconnecting the Layer 2 Circuit with the Layer 3 VPN

#### Step-by-Step Procedure

Before you can configure the logical tunnel interface in an MX Series router, you must create the tunnel services interface to be used for tunnel services.

1. Create the tunnel service interface on Router PE3. Include the **bandwidth** statement at the **[edit chassis fpc slot-number pic slot-number tunnel-services]** hierarchy level and specify the amount of bandwidth to reserve for tunnel services in gigabits per second.

```

[edit chassis]
fpc 1 {
    pic 1 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}

```

2. On Router PE3, configure the **lt-1/1/10** logical tunnel interface unit 0.

Router PE3 is the router that is *stitching* the Layer 2 circuit to the Layer 3 VPN using the logical tunnel interface. The configuration of the peer unit interfaces is what makes the interconnection.

Include the **encapsulation** statement and specify the **ethernet-ccc** option. Include the **peer-unit** statement and specify the logical interface unit 1 as the peer tunnel interface. Include the **family** statement and specify the **ccc** option.

Configure the **lt-1/1/10** logical interface unit 1 with **ethernet** encapsulation. Include the **peer-unit** statement and specify the logical interface unit 0 as the peer tunnel interface. Include the **family** statement and specify the **inet** option. Also include the **address** statement and specify **70.70.70.1/24** as the IPv4 address of the interface.



**NOTE:** The peering logical interfaces must belong to the same logical tunnel interface derived from the Tunnel Services PIC.

```

[edit interfaces]
lt-1/1/10 {
    unit 0 {

```

```

        encapsulation ethernet-ccc;
        peer-unit 1;
        family ccc;
    }
    unit 1 {
        encapsulation ethernet;
        peer-unit 0;
        family inet {
            address 70.70.70.1/24;
        }
    }
}

```

3. On each router, commit the configuration.

```

user@host> commit check
configuration check succeeds
user@host> commit

```

## Verifying the Layer 2 Circuit to Layer 3 VPN Interconnection

To verify that the interconnection is working properly, perform these tasks:

- [Verifying That the Layer 2 Circuit Connection to Router PE3 is Up on page 505](#)
- [Verifying LDP Neighbors and Targeted LDP LSPs on Router PE2 on page 506](#)
- [Verifying the Layer 2 Circuit Routes on Router PE2 on page 506](#)
- [Verifying That the Layer 2 Circuit Connection to Router PE2 is Up on page 507](#)
- [Verifying LDP Neighbors and Targeted LDP LSPs on Router PE3 on page 508](#)
- [Verifying a BGP Peer Session with the Route Reflector on Router PE3 on page 508](#)
- [Verifying the Layer 3 VPN Routes on Router PE3 on page 509](#)
- [Verifying the Layer 2 Circuit Routes on Router PE3 on page 509](#)
- [Verifying the MPLS Routes on Router PE3 on page 510](#)
- [Verifying Traffic Flow Between Router CE2 and Router CE3 on page 511](#)
- [Verifying Traffic Flow Between Router CE2 and Router CE5 on page 511](#)

### Verifying That the Layer 2 Circuit Connection to Router PE3 is Up

**Purpose** To verify that the Layer 2 circuit connection from Router PE2 to Router PE3 is **Up**. To also document the incoming and outgoing LDP labels and the circuit ID used by this Layer 2 circuit connection.

**Action** Verify that the Layer 2 circuit connection is up, using the **show l2circuit connections** command.

```
user@PE2> show l2circuit connections
```

Legend for connection status (St)

EI -- encapsulation invalid	NP -- interface h/w not present
MM -- mtu mismatch	Dn -- down
EM -- encapsulation mismatch	VC-Dn -- Virtual circuit Down
CM -- control-word mismatch	Up -- operational

VM -- vlan id mismatch                      CF -- Call admission control failure  
 OL -- no outgoing label                      IB -- TDM incompatible bitrate  
 NC -- intf encaps not CCC/TCC              TM -- TDM misconfiguration  
 BK -- Backup Connection                    ST -- Standby Connection  
 CB -- rcvd cell-bundle size bad           SP -- Static Pseudowire  
 LD -- local site signaled down           RS -- remote site standby  
 RD -- remote site signaled down          XX -- unknown

#### Legend for interface status

Up -- operational

Dn -- down

Neighbor: 3.3.3.3

Interface	Type	St	Time last up	# Up trans
ge-1/0/2.0(vc 100)	rmt	Up	Jan 7 02:14:13 2010	1

Remote PE: 3.3.3.3, Negotiated control-word: No  
 Incoming label: 301488, Outgoing label: 315264  
 Negotiated PW status TLV: No  
 Local interface: ge-1/0/2.0, Status: Up, Encapsulation: ETHERNET

**Meaning** The output shows that the Layer 2 circuit connection from Router PE2 to Router PE3 is **Up** and the connection is using the **ge-1/0/2.0** interface. Note that the outgoing label is **315264** and the incoming label is **301488**, the virtual circuit (VC) identifier is **100** and the encapsulation is **ETHERNET**.

### Verifying LDP Neighbors and Targeted LDP LSPs on Router PE2

**Purpose** To verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors.

**Action** Verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors, using the **show ldp neighbor** command.

```

user@PE2> show ldp neighbor
Address          Interface      Label space ID      Hold time
3.3.3.3          lo0.0          3.3.3.3:0           38
  
```

**Meaning** The output shows that Router PE2 has an LDP neighbor with the IPv4 address of **3.3.3.3**. Address 3.3.3.3 is the lo0.0 interface address of Router PE3. Notice that Router PE2 uses the local **lo0.0** interface for the LSP.

Verifying that the routers are LDP neighbors also verifies that the targeted LSP is established.

### Verifying the Layer 2 Circuit Routes on Router PE2

**Purpose** To verify that Router PE2 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE3.

**Action** Verify that Router PE2 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE3, using the **show route table mpls.0** command.

```

user@PE2> show route table mpls.0
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
  
```

```

0          *[MPLS/0] 1w3d 05:24:11, metric 1
           Receive
1          *[MPLS/0] 1w3d 05:24:11, metric 1
           Receive
2          *[MPLS/0] 1w3d 05:24:11, metric 1
           Receive
300560     *[LDP/9] 16:12:23, metric 1
           > to 10.10.2.1 via xe-0/1/0.0, Pop
300560(S=0) *[LDP/9] 16:12:23, metric 1
           > to 10.10.2.1 via xe-0/1/0.0, Pop
301008     *[LDP/9] 16:12:23, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Swap 299856
301488     *[L2CKT/7] 11:07:28
           > via ge-1/0/2.0, Pop
301536     *[LDP/9] 16:12:23, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Pop
301536(S=0) *[LDP/9] 16:12:23, metric 1
           > to 10.10.4.2 via xe-0/3/0.0, Pop
301712     *[LDP/9] 12:41:22, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Swap 315184
301728     *[LDP/9] 12:41:22, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Pop
301728(S=0) *[LDP/9] 12:41:22, metric 1
           > to 10.10.5.2 via xe-0/2/0.0, Pop
ge-1/0/2.0 *[L2CKT/7] 11:07:28, metric2 1
           > to 10.10.5.2 via xe-0/2/0.0, Push 315264

```

**Meaning** The output shows that Router PE2 pushes the **315264** outgoing label on the **L2CKT** route going out interface **ge-1/0/2.0**. The output also shows that Router PE2 pops the **301488** incoming label on the **L2CKT** coming from interface **ge-1/0/2.0**

### Verifying That the Layer 2 Circuit Connection to Router PE2 is Up

**Purpose** To verify that the Layer 2 circuit connection from Router PE3 to Router PE2 is **Up**, To also document the incoming and outgoing LDP labels and the circuit ID used by this Layer 2 circuit connection.

**Action** Verify that the Layer 2 circuit connection is up, using the **show l2circuit connections** command.

```
user@PE3> show l2circuit connections
```

Layer-2 Circuit Connections:

Legend for connection status (St)

EI -- encapsulation invalid	NP -- interface h/w not present
MM -- mtu mismatch	Dn -- down
EM -- encapsulation mismatch	VC-Dn -- Virtual circuit Down
CM -- control-word mismatch	Up -- operational
VM -- vlan id mismatch	CF -- Call admission control failure
OL -- no outgoing label	IB -- TDM incompatible bitrate
NC -- intf encaps not CCC/TCC	TM -- TDM misconfiguration
BK -- Backup Connection	ST -- Standby Connection
CB -- rcvd cell-bundle size bad	XX -- unknown

Legend for interface status

Up -- operational

Dn -- down

Neighbor: 2.2.2.2

```

Interface                Type St      Time last up          # Up trans
lt-1/1/10.0(vc 100)      rmt  Up      Jan  7 02:15:03 2010          1
Remote PE: 2.2.2.2, Negotiated control-word: No
Incoming label: 315264, Outgoing label: 301488
Local interface: lt-1/1/10.0, Status: Up, Encapsulation: ETHERNET

```

**Meaning** The output shows that the Layer 2 circuit connection from Router PE3 to Router PE2 is **Up** and the connection is using the logical tunnel (lt) interface. Note that the incoming label is **315264** and the outgoing label is **301488**, the virtual circuit (VC) identifier is **100**, and that the encapsulation is **ETHERNET**.

### Verifying LDP Neighbors and Targeted LDP LSPs on Router PE3

**Purpose** To verify that Router PE3 has a targeted LDP LSP to Router PE2 and that Router PE3 and Router PE2 are LDP neighbors.

**Action** Verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors, using the **show ldp neighbor** command.

```

user@PE2> show ldp neighbor
Address          Interface      Label space ID      Hold time
2.2.2.2          lo0.0          2.2.2.2:0           43
4.4.4.4          lo0.0          4.4.4.4:0           33

```

**Meaning** The output shows that Router PE3 has an LDP neighbor with the IPv4 address of **2.2.2.2**. Address 2.2.2.2 is the lo0.0 interface address of Router PE2. The output also shows that the interface used on Router PE3 for the LSP is **lo0.0**. Verifying that the routers are LDP neighbors also verifies that the targeted LSP is established.

### Verifying a BGP Peer Session with the Route Reflector on Router PE3

**Purpose** To verify that Router PE3 has a peer session established with the route reflector.

**Action** Verify that Router PE3 has a peer session established with the route reflector, using the **show bgp summary** command.

```

user@PE2> show bgp summary

```

```

Groups: 2 Peers: 2 Down peers: 0
Table      Tot Paths  Act Paths Suppressed  History  Damp State   Pending
bgp.13vpn.0      1          1          0          0          0          0
Peer          AS      InPkt    OutPkt    OutQ    Flaps  Last Up/Dwn
State|#Active/Received/Accepted/Damped...
7.7.7.7        65000     1597     1612      0        1    12:03:21 Establ
  bgp.12vpn.0: 0/0/0/0
  bgp.13vpn.0: 1/1/1/0
  L3VPN.inet.0: 1/1/1/0

```

**Meaning** The output shows that Router PE3 has a peer session with the router with the IPv4 address of **7.7.7.7**. Address 7.7.7.7 is the lo0.0 interface address of the route reflector. The output also shows that the peer session state is **Establ**, meaning that the session is established.

### Verifying the Layer 3 VPN Routes on Router PE3

**Purpose** To verify that Router PE3 has Layer 3 VPN routes to Router CE2, Router CE3, and Router CE5.

**Action** Verify that Router PE3 has routes to Router CE2, Router CE3, and Router CE5 in the Layer 3 VPN route table, using the **show route table L3VPN.inet.0** command. In this example, **L3VPN** is the name configured for the routing instance.

```
user@PE3> show route table L3VPN.inet.0
L3VPN.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

70.70.70.0/24      * [Direct/0] 11:13:59
                  > via lt-1/1/10.1
70.70.70.1/32     * [Local/0] 11:13:59
                  Local via lt-1/1/10.1
80.80.80.0/24     * [BGP/170] 11:00:41, localpref 100, from 7.7.7.7
                  AS path: I
                  > to 10.10.6.2 via xe-2/1/0.0, Push 16
90.90.90.0/24     * [Direct/0] 11:54:41
                  > via ge-1/0/1.0
90.90.90.1/32     * [Local/0] 11:54:41
                  Local via ge-1/0/1.0
```

**Meaning** The output shows that Router PE3 has a route to the IPv4 subnetwork address of **70.70.70.0**. Address 70.70.70.2 is the interface address of Router CE2. The output shows that Router PE3 has a route to the IPv4 subnetwork address of **80.80.80.0**. Address 80.80.80.2 is the interface address of Router CE5. The output shows that Router PE3 has a route to the IPv4 subnetwork address of **90.90.90.0**. Address 90.90.90.2 is the interface address of Router CE3.

### Verifying the Layer 2 Circuit Routes on Router PE3

**Purpose** To verify that Router PE3 has a route to Router PE2 in the Layer 2 circuit route table.

**Action** Verify that Router PE3 has a route to Router PE2 in the Layer 2 circuit route table, using the **show route table l2circuit.0** command.

```
user@PE3> show route table l2circuit.0
2.2.2.2:NoCtrlWord:5:100:Local/96 (1 entry, 1 announced)
  *L2CKT Preference: 7
    Next hop type: Indirect
    Next-hop reference count: 1
    Next hop type: Router
    Next hop: 10.10.5.1 via xe-2/2/0.0, selected
    Protocol next hop: 2.2.2.2
    Indirect next hop: 8cae0a0 -
    State: <Active Int>
    Local AS: 65000
    Age: 11:16:50 Metric2: 1
    Task: l2 circuit
    Announcement bits (1): 0-LDP
    AS path: I
    VC Label 315264, MTU 1500
```

**Meaning** The output shows that Router PE3 has a route to the IPv4 address of **2.2.2.2**. Address 2.2.2.2 is the lo0.0 interface address of Router PE2. Note that the VC label is **315264**. This label is the same as the incoming MPLS label displayed using the **show l2circuit connections** command.

### Verifying the MPLS Routes on Router PE3

**Purpose** To verify that Router PE3 has a route to Router PE2 in the MPLS route table.

**Action** Verify Router PE3 has a route to Router PE2 in the MPLS route table, using the **show route table mpls.0** command.

```
user@PE3> show route table mpls.0
mpls.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0          *[MPLS/0] 1w3d 05:29:02, metric 1
           Receive
1          *[MPLS/0] 1w3d 05:29:02, metric 1
           Receive
2          *[MPLS/0] 1w3d 05:29:02, metric 1
           Receive
16         *[VPN/0] 12:22:45
           to table L3VPN.inet.0, Pop
315184     *[LDP/9] 12:45:14, metric 1
           > to 10.10.20.1 via xe-2/0/0.0, Pop
315184(S=0) *[LDP/9] 12:45:14, metric 1
           > to 10.10.20.1 via xe-2/0/0.0, Pop
315200     *[LDP/9] 00:03:53, metric 1
           > to 10.10.20.1 via xe-2/0/0.0, Swap 625297
           to 10.10.6.2 via xe-2/1/0.0, Swap 299856
315216     *[LDP/9] 12:45:14, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, Pop
315216(S=0) *[LDP/9] 12:45:14, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, Pop
315232     *[LDP/9] 12:45:06, metric 1
           > to 10.10.1.1 via xe-2/3/0.0, Pop
315232(S=0) *[LDP/9] 12:45:06, metric 1
           > to 10.10.1.1 via xe-2/3/0.0, Pop
315248     *[LDP/9] 12:45:14, metric 1
           > to 10.10.5.1 via xe-2/2/0.0, Pop
315248(S=0) *[LDP/9] 12:45:14, metric 1
           > to 10.10.5.1 via xe-2/2/0.0, Pop
315264     *[L2CKT/7] 11:11:20
           > via lt-1/1/10.0, Pop
315312     *[RSVP/7] 11:26:01, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
315312(S=0) *[RSVP/7] 11:26:01, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
315328     *[RSVP/7] 11:26:01, metric 1
           > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
315360     *[RSVP/7] 11:26:01, metric 1
           > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
316208     *[RSVP/7] 00:03:32, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, label-switched-path
Bypass->10.10.9.1
316208(S=0) *[RSVP/7] 00:03:32, metric 1
           > to 10.10.6.2 via xe-2/1/0.0, label-switched-path
```

```

Bypass->10.10.9.1
lt-1/1/10.0      *[L2CKT/7] 11:11:20, metric2 1
                  > to 10.10.5.1 via xe-2/2/0.0, Push 301488

```

**Meaning** The output shows that Router PE3 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE2. Notice that the **301488** label is the same as the outgoing label displayed on Router PE2 using the **show l2circuit connections** command.

### Verifying Traffic Flow Between Router CE2 and Router CE3

**Purpose** To verify that the CE routers can send and receive traffic across the interconnection.

**Action** Verify that Router CE2 can send traffic to and receive traffic from Router CE3 across the interconnection, using the **ping** command.

```

user@CE2>ping 90.90.90.2
PING 90.90.90.2 (90.90.90.2): 56 data bytes
64 bytes from 90.90.90.2: icmp_seq=0 ttl=63 time=0.708 ms
64 bytes from 90.90.90.2: icmp_seq=1 ttl=63 time=0.610 ms

```

**Meaning** The output shows that Router CE2 can send an ICMP request to and receive a response from Router CE3 across the interconnection.

### Verifying Traffic Flow Between Router CE2 and Router CE5

**Purpose** To verify that the CE routers can send and receive traffic across the interconnection.

**Action** Verify that Router CE2 can send traffic to and receive traffic from Router CE5 across the interconnection, using the **ping** command.

```

user@CE2>ping 80.80.80.2
PING 80.80.80.2 (80.80.80.2): 56 data bytes
64 bytes from 80.80.80.2: icmp_seq=0 ttl=62 time=0.995 ms
64 bytes from 80.80.80.2: icmp_seq=1 ttl=62 time=1.005 ms

```

**Meaning** The output shows that Router CE2 can send an ICMP request to and receive a response from Router CE5 across the interconnection.

**Related Documentation**

- [Layer 3 VPN Overview on page 466](#)
- [Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN on page 468](#)



## CHAPTER 7

# Summary of Layer 3 VPN Configuration Statements

- `advertise-mode` (MPLS) on page 515
- `arp-prefix-limit` (Host Fast Reroute) on page 516
- `as-path-compare` on page 517
- `chained-composite-next-hop` on page 518
- `classifiers` on page 519
- `description` (Routing Instances) on page 519
- `domain-id` on page 520
- `domain-vpn-tag` on page 520
- `dynamic-tunnels` on page 521
- `egress-protection` (MPLS) on page 522
- `egress-protection` (Routing Instances) on page 523
- `egress-protection` (BGP) on page 524
- `extended-space` on page 525
- `forwarding-table` on page 526
- `global-arp-prefix-limit` (Host Fast Reroute) on page 527
- `global-supplementary-blackout-timer` (Host Fast Reroute) on page 529
- `group-address` (Routing Instances VPN) on page 531
- `host-fast-reroute` on page 532
- `independent-domain` on page 533
- `ingress` (Chained Composite Next Hop) on page 534
- `inet6` on page 535
- `inet6-vpn` on page 536
- `interface` (Host Fast Reroute) on page 537
- `l3vpn` on page 538
- `l3vpn` (transit) on page 539
- `label` on page 539

- [labeled-bgp](#) on page 540
- [link-protection \(Host Fast Reroute\)](#) on page 540
- [maximum-paths](#) on page 541
- [maximum-prefixes](#) on page 543
- [metric \(Protocols OSPF Sham Link\)](#) on page 544
- [multihop](#) on page 545
- [multipath \(Routing Options\)](#) on page 546
- [no-vrf-advertise](#) on page 547
- [no-vrf-propagate-ttl](#) on page 548
- [protect core](#) on page 549
- [protection \(Protocols BGP\)](#) on page 549
- [routing-instances \(Class of Service\)](#) on page 550
- [sham-link](#) on page 551
- [sham-link-remote](#) on page 552
- [source-class-usage](#) on page 553
- [supplementary-blackout-timer \(Host Fast Reroute\)](#) on page 554
- [vpn-unequal-cost](#) on page 555
- [vrf-propagate-ttl](#) on page 556
- [vrf-table-label](#) on page 557

## advertise-mode (MPLS)

<b>Syntax</b>	advertise-mode (stub-alias   stub-proxy);
<b>Hierarchy Level</b>	<p>[edit logical-systems <i>logical-system-name</i> protocols mpls egress-protection context-identifier <i>context-id</i>],</p> <p>[edit logical-systems <i>logical-system-name</i> protocols mpls label-switched-path <i>lsp-name</i> egress-protection context-identifier <i>context-id</i>],</p> <p>[edit protocols mpls egress-protection context-identifier <i>context-id</i>],</p> <p>[edit protocols mpls label-switched-path <i>lsp-name</i> egress-protection context-identifier <i>context-id</i>]</p>
<b>Release Information</b>	Statement introduced in Junos OS Release 13.3.
<b>Description</b>	<p>Configure the method for the interior gateway protocol (IGP) to advertise egress protection availability.</p> <p>Egress protection availability is advertised in the IGP. Label protocols along with CSPF use this information to do egress protection.</p>
<b>Options</b>	<p><b>stub-alias</b>—Context identifier has an alias.</p> <p>In the alias method, the LSP end-point address has an explicit backup egress node where the backup node can be learned or configured on the penultimate hop node (PHN) of a protected LSP. With this model, the PHN of a protected LSP sets up the bypass LSP tunnel to back up the egress node by avoiding the primary egress node. This model requires a Junos OS upgrade in core nodes, but is flexible enough to support all traffic engineering constraints.</p> <p><b>stub-proxy</b>—Context-identifier has a stub proxy node.</p> <p>A stub node is one that only appears at the end of an AS path, which means it does not provide transit service. In this mode, known as the virtual or proxy mode, the LSP end-point address is represented as a node with bidirectional links, with the LSP's primary egress node and backup egress node. With this representation, the penultimate hop of the LSP primary egress point can behave like a PLR in setting up a bypass tunnel to back up the egress by avoiding the primary egress node. This model has the advantage that you do not need to upgrade Junos OS on core nodes and thereby helps operators to deploy this technology.</p>
<b>Required Privilege Level</b>	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Egress Protection for Layer 3 VPN Edge Protection Overview on page 269</a></li> <li>• <a href="#">Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP on page 283</a></li> </ul>

## arp-prefix-limit (Host Fast Reroute)

<b>Syntax</b>	<code>arp-prefix-limit <i>number</i>;</code>
<b>Hierarchy Level</b>	<code>[edit logical-systems <i>logical-system-name</i> routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>],</code> <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code>
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Override the ARP prefix limit for the specified host fast-reroute (HFRR) profile.</p> <p>When you configure HFRR, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices.</p> <p>There are two configuration statements (<a href="#">global-arp-prefix-limit</a> and <a href="#">arp-prefix-limit</a>) that set the ARP prefix limit, one at the global <code>[edit routing-options host-fast-reroute]</code> hierarchy level and the other at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, respectively. The global <a href="#">global-arp-prefix-limit</a> statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The <a href="#">arp-prefix-limit</a> statement overrides the <a href="#">global-arp-prefix-limit</a> for that HFRR profile for that protected interface.</p> <p>Warning system log messages begin when the ARP routes in an HFRR profile reaches 80% of the configured limit. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.</p> <p>After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.</p> <p>There are global and per-HFRR CLI statements (<a href="#">global-supplementary-blackout-timer</a> and <a href="#">supplementary-blackout-timer</a>) to configure the supplementary blackout timer. The global value is at the <code>[edit routing-options host-fast-reroute]</code> hierarchy level and applies to all HFRR profiles on the routing device. The value for the routing-instance interface is at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, and overrides the global value for that HFRR profile only.</p> <p>When the blackout timer expires, the HFRR profile is reactivated, and the Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.</p> <p>If an HFRR profile is in the deactivated state, a reevaluation of the ARP state is preformed during every commit operation or whenever the routing process (rpd) is restarted with the <code>restart routing</code> command.</p>
<b>Default</b>	If you omit the <a href="#">arp-prefix-limit</a> statement, the <a href="#">global-arp-prefix-limit</a> takes effect for all HFRR profiles on the device. If you omit both of these statements, there is no ARP prefix limit for host fast reroute.

<b>Options</b>	<i>number</i> —Maximum number of ARP HFRR routes allowed. <b>Range:</b> 1 through 10,000 HFRR routes
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Link Protection with Host Fast Reroute on page 242</a></li></ul>

---

## as-path-compare

---

<b>Syntax</b>	as-path-compare;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options multipath], [edit routing-instances <i>routing-instance-name</i> routing-options multipath]
<b>Release Information</b>	Statement introduced in Junos OS Release 10.1. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	Specify to have the algorithm that is used to determine the active path compare the AS numbers in the AS path. In a VPN scenario with multiple BGP paths, the algorithm selects as the active path the route whose AS numbers match. By default, the algorithm evaluates only the length and not the contents of the AS path.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes on page 97</a></li></ul>

## chained-composite-next-hop

---

<b>Syntax</b>	chained-composite-next-hop { <a href="#">ingress</a> ; transit; }
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options <a href="#">forwarding-table</a> ], [edit routing-options <a href="#">forwarding-table</a> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.1. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	<p>Allows you to configure the chained composite next hops for devices handling ingress or transit traffic in the network.</p> <p>Enable or disable composite-chained next hops for various traffic types.</p> <p>The remaining statements are explained separately.</p>
<b>Default</b>	This statement is disabled by default.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li><li>• <a href="#">forwarding-table on page 526</a></li></ul>

## classifiers

<b>Syntax</b>	<code>classifiers {     exp (<i>classifier-name</i>   default); }</code>
<b>Hierarchy Level</b>	[edit class-of-service routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4.
<b>Description</b>	For routing instances with VRF table labels enabled, apply a custom MPLS EXP classifier to the routing instance. You can apply the default MPLS EXP classifier or one that is previously defined.
<b>Default</b>	If you do not include this statement, the default MPLS EXP classifier is applied to the routing instance.
<b>Options</b>	<i>classifier-name</i> —Name of the behavior aggregate MPLS EXP classifier.
<b>Required Privilege Level</b>	interface—To view this statement in the configuration. interface-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs on page 58</a></li> <li>• <i>Junos OS Network Interfaces Library for Routing Devices</i></li> </ul>

## description (Routing Instances)

<b>Syntax</b>	<code>description text;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> ], [edit routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Statement introduced in Junos OS Release 11.1 for EX Series switches. Statement introduced in Junos OS Release 11.3 for the QFX Series. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	Provide a text description for the routing instance. If the text includes one or more spaces, enclose it in quotation marks (" "). Any descriptive text you include is displayed in the output of the <b>show route instance detail</b> command and has no effect on the operation of the routing instance.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <i>Configuring Routing Instances on PE Routers in VPNs</i></li> <li>• <i>show route instance</i></li> </ul>

## domain-id

---

<b>Syntax</b>	<code>domain-id <i>domain-id</i>;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols (ospf   ospf3)], [edit routing-instances <i>routing-instance-name</i> protocols (ospf   ospf3)]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Statement introduced in Junos OS Release 9.0 for EX Series switches.
<b>Description</b>	Specify a domain ID for a route. The domain ID identifies the OSPF domain from which the route originated.
<b>Options</b>	<b><i>domain-id</i></b> —You can specify either an IP address or an IP address and a local identifier using the following format: <i>ip-address:local-identifier</i> . If you do not specify a local identifier with the IP address, the identifier is assumed to have a value of 0. <b>Default:</b> If the router ID is not configured in the routing instance, the router ID is derived from an interface address belonging to the routing instance.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Configuring Routing Between PE and CE Routers in Layer 3 VPNs on page 31</a></li></ul>

## domain-vpn-tag

---

<b>Syntax</b>	<code>domain-vpn-tag <i>number</i>;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols (ospf   ospf3)], [edit routing-instances <i>routing-instance-name</i> protocols (ospf   ospf3)]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Statement introduced in Junos OS Release 9.0 for EX Series switches.
<b>Description</b>	Set a virtual private network (VPN) tag for OSPFv2 external routes generated by the provider edge (PE) routing device.
<b>Options</b>	<b><i>number</i></b> —VPN tag.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Configuring Routing Between PE and CE Routers in Layer 3 VPNs on page 31</a></li></ul>

## dynamic-tunnels

<b>Syntax</b>	<pre>dynamic-tunnels <i>tunnel-name</i> {   destination-networks <i>prefix</i>;   gre;   rsvp-te <i>entry-name</i> {     destination-networks <i>network-prefix</i>;     label-switched-path-template {       default-template;       <i>template-name</i>;     }   }   source-address <i>address</i>; }</pre>
<b>Hierarchy Level</b>	<p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-options],</p> <p>[edit routing-instances <i>routing-instance-name</i> routing-options],</p> <p>[edit routing-options]</p>
<b>Release Information</b>	<p>Statement introduced before Junos OS Release 7.4.</p> <p>Statement introduced in Junos OS Release 12.3 for ACX Series routers.</p>
<b>Description</b>	Configure a dynamic tunnel between two PE routers.
<b>Options</b>	<p><b><i>tunnel-name</i></b>—Name of the dynamic tunnel.</p> <p>The remaining statements are explained separately.</p>
<b>Required Privilege Level</b>	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <i>Example: Configuring a Two-Tiered Virtualized Data Center for Large Enterprise Networks</i></li> <li>• <a href="#">Configuring GRE Tunnels for Layer 3 VPNs on page 86</a></li> <li>• <i>Configuring Dynamic Tunnels</i></li> </ul>

## egress-protection (MPLS)

---

<b>Syntax</b>	<pre>egress-protection {     context-identifier <i>context-id</i> {         primary   protector;         metric <i>igp-metric-value</i>;         advertise-mode (stub-alias   stub-proxy);     } }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> protocols mpls], [edit logical-systems <i>logical-system-name</i> protocols mpls label-switched-path <i>lsp-name</i> ], [edit protocols mpls], [edit protocols mpls label-switched-path <i>lsp-name</i> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 10.4. Options <b>primary</b> , <b>protector</b> , and <b>metric</b> introduced in Junos OS Release 11.4R3. Option <b>advertise-mode</b> introduced in Junos OS Release 13.3.
<b>Description</b>	Enables an Edge Protection Virtual Circuit (EPVC) for the MPLS protocol.
<b>Options</b>	<b>context-identifier <i>context-id-ip-address</i></b> —(Optional) The context identifier IPv4 address.  <b>metric <i>igp-metric-value</i></b> —(Optional) The IGP metric value ranging from 2 through 16777215.  <b>(primary   protector)</b> —On the primary PE router, configure as type <b>primary</b> . On the protector PE router, configure as type <b>protector</b> .
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Egress Protection for Layer 3 VPN Services on page 275</a></li><li>• <a href="#">Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP on page 283</a></li></ul>

## egress-protection (Routing Instances)

---



<b>Syntax</b>	<pre>egress-protection {   context-identifier <i>context-id-ip-address</i>; }</pre>
<b>Hierarchy Level</b>	[edit routing-instances <i>routing-instance-name</i> ].
<b>Release Information</b>	Statement introduced in Junos OS Release 11.4R3.
<b>Description</b>	Enable egress instance protection and provides customer edge (CE) VRF-level context-id granularity for each virtual routing and forwarding (VRF) instance.
<b>Options</b>	<b>context-identifier <i>context-id-ip-address</i></b> —(Optional) The IPv4 address for the context identifier of the protected PE router.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Egress Protection for Layer 3 VPN Services on page 275</a></li></ul>

## egress-protection (BGP)

---

<b>Syntax</b>	<pre>egress-protection {     context-identifier <i>context-id-ip-address</i>;     keep-import <i>policy-name</i>; }</pre>
<b>Hierarchy Level</b>	[edit protocols bgp group <i>group-name</i> family inet labeled-unicast], [edit protocols bgp group <i>group-name</i> family inet-vpn unicast], [edit protocols bgp group <i>group-name</i> family inet6-vpn unicast], [edit protocols bgp group <i>group-name</i> family iso-vpn unicast]
<b>Release Information</b>	Statement introduced in Junos OS Release 11.4R3. Statement introduced for labeled unicast in Junos OS Release 14.1.
<b>Description</b>	Enable egress protection for the configured BGP Network Layer Reachability Information (NLRI).
<b>Options</b>	<p><b>context-identifier <i>context-id-ip-address</i></b>—(Optional) The IPv4 address of the context identifier.</p> <p><b>keep-import <i>policy-name</i></b>—The import policy that contains routes with the configured route target community. BGP keeps all these routes in the VPN routing table. The protector PE router scans the policy and builds the remote instance with the configured community name from the policy.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Egress Protection for Layer 3 VPN Services on page 275</a></li><li>• <a href="#">Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP on page 283</a></li><li>• <a href="#">Example: Configuring Egress Protection for BGP Labeled Unicast on page 345</a></li></ul>

## extended-space

<b>Syntax</b>	extended-space;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options forwarding-table chained-composite-next-hop ingress <b>l3vpn</b> ], [edit routing-options forwarding-table chained-composite-next-hop ingress <b>l3vpn</b> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	Accept more than one million Layer 3 VPN BGP updates with unique inner VPN labels. The neighboring PE routers are typically from other vendors and configured to assign a unique inner label to each Layer 3 VPN BGP route.
<div>  <p><b>NOTE:</b> This statement is supported on Juniper Networks routers containing only MPCs and MICs and with chassis network services configured with the <b>enhanced-ip</b> setting.</p> </div>	
<div>  <p><b>WARNING:</b> You cannot configure the <b>extended-space</b> statement on a MX system with a MS-DPC unless you have also configured FIB localization with the MS-DPC set as a FIB-remote. This is because the MS-DPC cannot install nexthop in <b>extended-space</b> so the use of a default nexthop pointing to a FIB-local FPC is necessary to ensure proper data forwarding.</p> </div>	
<b>Default</b>	This statement is disabled by default.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li> </ul>

## forwarding-table

---

<b>Syntax</b>	<pre>forwarding-table {     chained-composite-next-hop;     export [ <i>policy-name</i> ];     (indirect-next-hop   no-indirect-next-hop);     (indirect-next-hop-change-acknowledgements           no-indirect-next-hop-change-acknowledgements);     krt-nexthop-ack-timeout <i>interval</i>;     unicast-reverse-path (active-paths   feasible-paths); }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options], [edit routing-options]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Statement introduced in Junos OS Release 9.0 for EX Series switches. Statement introduced in Junos OS Release 11.3 for the QFX Series.
<b>Description</b>	Configure information about the routing device's forwarding table.  The remaining statements are explained separately.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <i>Example: Load Balancing BGP Traffic</i></li></ul>

## global-arp-prefix-limit (Host Fast Reroute)

<b>Syntax</b>	<code>global-arp-prefix-limit <i>number</i>;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options host-fast-reroute], [edit routing-options host-fast-reroute]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Set the ARP prefix limit for all host fast-reroute (HFRR) profiles on the routing device.</p> <p>When you configure HFRR, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices.</p> <p>There are two configuration statements (<b>global-arp-prefix-limit</b> and <b>arp-prefix-limit</b>) that set the ARP prefix limit, one at the global [edit routing-options host-fast-reroute] hierarchy level and the other at the [edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>] hierarchy level, respectively. The global <b>global-arp-prefix-limit</b> statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The <b>arp-prefix-limit</b> statement overrides the <b>global-arp-prefix-limit</b> for that HFRR profile for that protected interface.</p> <p>Warning system log messages begin when the ARP routes in an HFRR profile reaches 80% of the configured limit. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.</p> <p>After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.</p> <p>There are global and per-HFRR CLI statements (<b>global-supplementary-blackout-timer</b> and <b>supplementary-blackout-timer</b>) to configure the supplementary blackout timer. The global value is at the [edit routing-options host-fast-reroute] hierarchy level and applies to all HFRR profiles on the routing device. The value for the routing-instance interface is at the [edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>] hierarchy level, and overrides the global value for that HFRR profile only.</p> <p>When the blackout timer expires, the HFRR profile is reactivated, and the Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.</p> <p>If an HFRR profile is in the deactivated state, a reevaluation of the ARP state is preformed during every commit operation or whenever the routing process (rpd) is restarted with the <b>restart routing</b> command.</p>
<b>Default</b>	If you omit the <b>arp-prefix-limit</b> statement, the <b>global-arp-prefix-limit</b> takes effect for all HFRR profiles on the device. If you omit both of these statements, there is no ARP prefix limit for host fast reroute.

**Options**    *number*—Maximum number of ARP HFRR routes allowed.  
                 **Range:** 1 through 10,000 HFRR routes

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

**Related**                • [Example: Configuring Link Protection with Host Fast Reroute on page 242](#)  
**Documentation**

## global-supplementary-blackout-timer (Host Fast Reroute)

<b>Syntax</b>	<code>global-supplementary-blackout-timer <i>minutes</i>;</code>
<b>Hierarchy Level</b>	<code>[edit logical-systems <i>logical-system-name</i> routing-options host-fast-reroute],</code> <code>[edit routing-options host-fast-reroute]</code>
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Set the blackout timer for all host fast-reroute (HFRR) profiles on the routing device.</p> <p>When you configure HFRR, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices.</p> <p>There are two configuration statements (<code>global-arp-prefix-limit</code> and <code>arp-prefix-limit</code>) that set the ARP prefix limit, one at the global <code>[edit routing-options host-fast-reroute]</code> hierarchy level and the other at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, respectively. The global <code>global-arp-prefix-limit</code> statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The <code>arp-prefix-limit</code> statement overrides the <code>global-arp-prefix-limit</code> for that HFRR profile for that protected interface.</p> <p>Warning system log messages begin when the ARP routes in an HFRR profile reaches 80% of the configured limit. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.</p> <p>After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.</p> <p>There are global and per-HFRR CLI statements (<code>global-supplementary-blackout-timer</code> and <code>supplementary-blackout-timer</code>) to configure the supplementary blackout timer. The global value is at the <code>[edit routing-options host-fast-reroute]</code> hierarchy level and applies to all HFRR profiles on the routing device. The value for the routing-instance interface is at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, and overrides the global value for that HFRR profile only.</p> <p>When the blackout timer expires, the HFRR profile is reactivated, and the Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.</p> <p>If an HFRR profile is in the deactivated state, a reevaluation of the ARP state is performed during every commit operation or whenever the routing process (rpd) is restarted with the <code>restart routing</code> command.</p>
<b>Default</b>	If you omit the <code>supplementary-blackout-timer</code> statement, the <code>global-supplementary-blackout-timer</code> takes effect for all HFRR profiles on the device. If you omit both of these statements, the blackout timeout value is set to the ARP cache


timeout value, which by default is 20 minutes and is configurable with the **aging-timer** statement at the **[edit system arp]** hierarchy level.

**Options**    **minutes**—Number of minutes, in addition to the ARP cache timeout value, before all HFRR profiles on the routing device are reactivated after the ARP prefix limit is exceeded.  
**Range:** 1 through 15 minutes

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

**Related Documentation**    • [Example: Configuring Link Protection with Host Fast Reroute on page 242](#)

## group-address (Routing Instances VPN)


<b>Syntax</b>	<code>group-address address;</code>
<b>Hierarchy Level</b>	<p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm family inet],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm family inet6],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm family inet],</p> <p>[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm family inet6],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm family inet],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-asm family inet6],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm family inet],</p> <p>[edit routing-instances <i>routing-instance-name</i> provider-tunnel pim-ssm family inet6]</p>
<b>Release Information</b>	<p>Statement introduced before Junos OS Release 7.4.</p> <p>Statement name changed (from <b>vpn-group-address</b> to <b>group-address</b>) and hierarchy levels changed in Junos OS Release 11.4.</p>
<b>Description</b>	Specify a group address on which to encapsulate multicast traffic from a virtual private network (VPN) instance.
<div>  <p><b>NOTE:</b> IPv6 provider tunnels are not currently supported for draft-rosen MVPNs. They are supported for MBGP MVPNs.</p> </div>	
<b>Options</b>	<p><b>address</b>—For IPv4, IP address whose high-order bits are 1110, giving an address range from 224.0.0.0 through 239.255.255.255, or simply 224.0.0.0/4. For IPv6, IP address whose high-order bits are FF00 (FF00::/8).</p>
<b>Required Privilege Level</b>	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <i>Example: Configuring Any-Source Multicast for Draft-Rosen VPNs</i></li> <li>• <a href="#">Configuring Multicast Layer 3 VPNs on page 84</a></li> <li>• <i>Multicast Protocols Feature Guide for Routing Devices</i></li> </ul>

## host-fast-reroute

---

<b>Syntax</b>	<pre>host-fast-reroute {     global-arp-prefix-limit <i>number</i>;     global-supplementary-blackout-timer <i>minutes</i>; }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options], [edit routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Configure global settings for all host fast reroute (HFRR) profiles configured on a routing device.</p> <p>The remaining statements are described separately.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Link Protection with Host Fast Reroute on page 242</a></li></ul>

## independent-domain

<b>Syntax</b>	<code>independent-domain &lt;no-attrset&gt;;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options autonomous-system <i>autonomous-system</i> ], [edit routing-instances <i>routing-instance-name</i> routing-options autonomous-system <i>autonomous-system</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. <b>no-attrset</b> option introduced in Junos OS Release 10.4.
<b>Description</b>	<p>Configure an independent AS domain.</p> <p>The independent domain uses transitive path attribute 128 (attribute set) messages to tunnel the independent domain's BGP attributes through the internal BGP (IBGP) core.</p> <p>This improves the transparency of Layer 3 VPN services for customer networks by preventing the IBGP routes that originate within an autonomous system (AS) in the customer network from being sent to a service provider's AS. Similarly, IBGP routes that originate within an AS in the service provider's network are prevented from being sent to a customer AS.</p>
<div style="display: flex; align-items: center;">  <div> <p><b>NOTE:</b> In Junos OS Release 10.3 and later, if BGP receives attribute 128 and you have not configured an independent domain in any routing instance, BGP treats the received attribute 128 as an unknown attribute.</p> </div> </div>	
<b>Options</b>	<b>no-attrset</b> —(Optional) Disables attribute set messages on the independent AS domain.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains on page 358</a></li> <li>• <a href="#">Configuring Layer 3 VPNs to Carry IBGP Traffic on page 49</a></li> <li>• <a href="#">autonomous-system</a></li> </ul>

## ingress (Chained Composite Next Hop)

<b>Syntax</b>	<pre>ingress {     (evpn   no-evpn);     (fec129-vpws   no-fec129-vpws);     (l2ckt   no-l2ckt);     (l2vpn   no-l2vpn);     l3vpn;     labeled-bgp; }</pre>
<b>Hierarchy Level</b>	<p>[edit logical-systems <i>logical-system-name</i> routing-options forwarding-table <a href="#">chained-composite-next-hop</a>],</p> <p>[edit routing-options forwarding-table <a href="#">chained-composite-next-hop</a>]</p>
<b>Release Information</b>	<p>This statement was introduced in Junos OS Release 12.1.</p> <p><b>labeled-bgp</b> option was introduced in Junos OS Release 12.3.</p> <p><b>l2ckt</b>, <b>l2vpn</b>, <b>no-l2ckt</b>, and <b>no-l2vpn</b> options were introduced in Junos OS Release 13.3.</p> <p><b>evpn</b>, <b>fec129-vpws</b>, <b>no-evpn</b>, and <b>no-fec129-vpws</b> options were introduced in Junos OS Release 14.1.</p>
<b>Description</b>	Allows you to configure chained composite next hops for devices handling transit traffic in the network.
<b>Default</b>	This statement is disabled by default.
<b>Options</b>	<p><b>evpn   no-evpn</b>—Enable or disable composite chained next hop for ingress EVPN label-switched paths (LSPs).</p> <p><b>fec129-vpws   no-fec129-vpws</b>—Enable or disable composite chained next hop for ingress FEC 129 virtual private wire service (VPWS) LSPs.</p> <p><b>l2ckt   no-l2ckt</b>—Enable or disable composite chained next hop for ingress Layer 2 circuit LSPs.</p> <p><b>l2vpn   no-l2vpn</b>—Enable or disable composite chained next hop for ingress Layer 2 virtual private network (VPN) LSPs.</p> <p>The remaining statements are explained separately.</p>
<b>Required Privilege Level</b>	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li> <li>• <a href="#">chained-composite-next-hop on page 518</a></li> </ul>

## inet6

---

<b>Syntax</b>	inet6;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options forwarding-table chained-composite-next-hop ingress <b>labeled-bgp</b> ], [edit routing-options forwarding-table chained-composite-next-hop ingress <b>labeled-bgp</b> ]
<b>Release Information</b>	This statement was introduced in Junos OS Release 12.3.
<b>Description</b>	Allows you to configure chained composite next hops for IPv6 labeled-unicast routes.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li></ul>

## inet6-vpn

---


<b>Syntax</b>	<pre>inet6-vpn (any   multicast   unicast) {     aggregate-label;     prefix-limit <i>maximum</i>;     rib-group <i>rib-group-name</i>; }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> protocols bgp family], [edit logical-systems <i>logical-system-name</i> protocols bgp group <i>group-name</i> family], [edit protocols bgp family], [edit protocols bgp group <i>group-name</i> family]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4.
<b>Description</b>	Enable IP version 6 (IPv6) on the provider edge (PE) router for the Layer 3 VPN.
<b>Options</b>	<p><b>any</b>—Configure the family type to be both multicast and unicast.</p> <p><b>multicast</b>—Configure the family type to be multicast. This means that the BGP peers are being used only to carry the unicast routes that are being used by multicast for resolving the multicast routes.</p> <p><b>prefix-limit <i>maximum</i></b>—Maximum prefix limit. <b>Range:</b> 1 through 4,294,967,295 <b>Default:</b> 1</p> <p><b>rib-group <i>rib-group-name</i></b>—The name of the routing table group.</p> <p><b>unicast</b>—Configure the family type to be unicast. This means that the BGP peers only carry the unicast routes that are being used for unicast forwarding purposes.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Configuring Layer 3 VPNs to Carry IPv6 Traffic on page 46</a></li><li>• <i>Junos OS Routing Protocols Library for Routing Devices</i></li></ul>

## interface (Host Fast Reroute)

---

<b>Syntax</b>	<pre>interface <i>interface-name</i> {     <i>arp-prefix-limit</i> <i>number</i>;     link-protection;     supplementary-blackout-timer <i>minutes</i>; }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options], [edit routing-instances <i>routing-instance-name</i> routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Configure host fast reroute settings, including link protection for the interface and an ARP prefix limit.</p> <p>The remaining statements are described separately.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Link Protection with Host Fast Reroute on page 242</a></li></ul>

## l3vpn

<b>Syntax</b>	l3vpn { <a href="#">extended-space</a> ; }
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options forwarding-table chained-composite-next-hop <a href="#">ingress</a> ], [edit routing-options forwarding-table chained-composite-next-hop <a href="#">ingress</a> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.1.
<b>Description</b>	(M120 routers, M320 routers with Enhanced III FPCs, MX Series routers, and T Series routers only) Accept larger numbers of Layer 3 VPN BGP updates with unique inner VPN labels (up to one million). The neighboring PE routers are typically from other vendors and are configured to assign a unique inner label to each Layer 3 VPN BGP route.
<div>  <b>NOTE:</b> <ul style="list-style-type: none"> <li>On MX Series routers, this statement is not supported when the router has both DPCs and MPCs installed. You must remove one or the other type of card to successfully use this statement.</li> <li>In Junos OS Release 11.4 and earlier, the l3vpn statement was titled as l3vpn-composite-nexthop, and the statement was available at the [edit logical-systems <i>logical-system-name</i> routing-options] and [edit routing-options] hierarchy levels.</li> </ul> </div>	
The remaining statement is explained separately.	
<b>Default</b>	This statement is disabled by default.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li><a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li> <li><a href="#">extended-space on page 525</a></li> </ul>

## l3vpn (transit)

---

<b>Syntax</b>	transit l3vpn;
<b>Hierarchy Level</b>	[edit logical-systems logical-system-name routing-options forwarding-table chained-composite-next-hop transit] [edit routing-options forwarding-table chained-composite-next-hop transit]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.1.
<b>Description</b>	Configure l3vpn settings on the transit router.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">chained-composite-next-hop on page 518</a></li> </ul>

## label

---

<b>Syntax</b>	label { allocation <i>label-allocation-policy</i> ; substitution <i>label-substitution-policy</i> ; }
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options] [edit routing-instances <i>routing-instance-name</i> routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 10.0.
<b>Description</b>	Specify label allocation and substitution policies on a per-route basis.
<b>Options</b>	<p><b>allocation</b> <i>label-allocation-policy</i>—Specify a policy to generate labels on a per-route basis.</p> <p><b>substitution</b> <i>label-substitution-policy</i>—Specify a policy to substitute labels on a per-route basis. The label substitution policy is used to determine whether or not a label should be substituted on an AS border router. The results of the policy operation are either accept (label substitution is performed) or reject (label substitution is not performed).</p> <p><b>Default:</b> accept</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Configuring a Label Allocation and Substitution Policy for VPNs on page 81</a></li> </ul>

## labeled-bgp

---

<b>Syntax</b>	labeled-bgp { <a href="#">inet6</a> ; }
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-options forwarding-table chained-composite-next-hop <a href="#">ingress</a> ], [edit routing-options forwarding-table chained-composite-next-hop <a href="#">ingress</a> ]
<b>Release Information</b>	This statement was introduced in Junos OS Release 12.3.
<b>Description</b>	Allows you to configure chained composite next hops for IPv6 labeled-unicast routes.  The other statement is explained separately.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 112</a></li></ul>

## link-protection (Host Fast Reroute)

---

<b>Syntax</b>	link-protection;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options interface <i>interface-name</i> ], [edit routing-instances <i>routing-instance-name</i> routing-options interface <i>interface-name</i> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	Enable link protection on a specified interface in a routing instance.  Configuring link protection on an interface ensures that traffic is rerouted to the destination device through alternate loop-free paths that traverse the protected interface.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Configuring Link Protection with Host Fast Reroute on page 242</a></li></ul>

## maximum-paths

<b>Syntax</b>	<code>maximum-paths <i>path-limit</i> &lt;log-interval <i>seconds</i>&gt; &lt;log-only   threshold <i>value</i>&gt;;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options], [edit logical-systems <i>logical-system-name</i> routing-options], [edit routing-instances <i>routing-instance-name</i> routing-options], [edit routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 8.0. Statement introduced in Junos OS Release 9.0 for EX Series switches. Statement introduced in Junos OS Release 11.3 for the QFX Series.
<b>Description</b>	Configure a limit for the number of routes installed in a routing table based upon the route path.



**NOTE:** The `maximum-paths` statement is similar to the `maximum-prefixes` statement. The `maximum-prefixes` statement limits the number of unique destinations in a routing instance. For example, suppose a routing instance has the following routes:

```
OSPF 10.10.10.0/24
ISIS 10.10.10.0/24
```

These are two routes, but only one destination (prefix). The `maximum-paths` limit applies the total number of routes (two). The `maximum-prefixes` limit applies to the total number of unique prefixes (one).

<b>Options</b>	<p><code>log-interval <i>seconds</i></code>—(Optional) Minimum time interval (in seconds) between log messages.</p> <p><b>Range:</b> 5 through 86,400</p> <p><code>log-only</code>—(Optional) Sets the route limit as an advisory limit. An advisory limit triggers only a warning, and additional routes are not rejected.</p> <p><code><i>path-limit</i></code>—Maximum number of routes. If this limit is reached, a warning is triggered and additional routes are rejected.</p> <p><b>Range:</b> 1 through 4,294,967,295 (<math>2^{32} - 1</math>)</p> <p><b>Default:</b> No default</p> <p><code>threshold <i>value</i></code>—(Optional) Percentage of the maximum number of routes that starts triggering a warning. You can configure a percentage of the <code><i>path-limit</i></code> value that starts triggering the warnings.</p> <p><b>Range:</b> 1 through 100</p>
----------------	---



NOTE: When the number of routes reaches the threshold value, routes are still installed into the routing table while warning messages are sent. When the number of routes reaches the *path-limit* value, then additional routes are rejected.

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

**Related Documentation**

- [Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs on page 44](#)

## maximum-prefixes

<b>Syntax</b>	<code>maximum-prefixes <i>prefix-limit</i> &lt;log-interval <i>seconds</i>&gt; &lt;log-only   threshold <i>percentage</i>&gt;;</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options], [edit logical-systems <i>logical-system-name</i> routing-options], [edit routing-instances <i>routing-instance-name</i> routing-options], [edit routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 8.0. Statement introduced in Junos OS Release 9.0 for EX Series switches. Statement introduced in Junos OS Release 11.3 for the QFX Series.
<b>Description</b>	Configure a limit for the number of routes installed in a routing table based upon the route prefix.  Using a prefix limit, you can curtail the number of prefixes received from a CE router in a VPN. Prefix limits apply only to dynamic routing protocols and are not applicable to static or interface routes.



**NOTE:** The `maximum-prefixes` statement is similar to the `maximum-paths` statement. The `maximum-prefixes` statement limits the number of unique destinations in a routing instance. For example, suppose a routing instance has the following routes:

```
OSPF 10.10.10.0/24
ISIS 10.10.10.0/24
```

These are two routes, but only one destination (prefix). The `maximum-paths` limit applies the total number of routes (two). The `maximum-prefixes` limit applies to the total number of unique prefixes (one).

<b>Options</b>	<p><code>log-interval <i>seconds</i></code>—(Optional) Minimum time interval (in seconds) between log messages.  <b>Range:</b> 5 through 86,400</p> <p><code>log-only</code>—(Optional) Sets the prefix limit as an advisory limit. An advisory limit triggers only a warning, and additional routes are not rejected.</p> <p><code><i>prefix-limit</i></code>—Maximum number of route prefixes. If this limit is reached, a warning is triggered and any additional routes are rejected.  <b>Range:</b> 1 through 4,294,967,295  <b>Default:</b> No default</p> <p><code>threshold <i>value</i></code>—(Optional) Percentage of the maximum number of prefixes that starts triggering a warning. You can configure a percentage of the <code><i>prefix-limit</i></code> value that starts triggering the warnings.</p>
----------------	---

**Range:** 1 through 100



**NOTE:** When the number of routes reaches the threshold value, routes are still installed into the routing table while warning messages are sent. When the number of routes reaches the *prefix-limit* value, then additional routes are rejected.

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


**Related Documentation**

- [Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs on page 44](#)

## metric (Protocols OSPF Sham Link)

<b>Syntax</b>	<i>metric number</i> ;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols ospf area <i>area-id</i> sham-link-remote <i>address</i> ], [edit routing-instances <i>routing-instance-name</i> protocols ospf area <i>area-id</i> sham-link-remote <i>address</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4.
<b>Description</b>	Specify the cost of using the OSPF sham link.
<b>Default</b>	<i>number</i> —1
<b>Options</b>	<i>number</i> —1 through 65,535
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	• <a href="#">Example: Configuring OSPFv2 Sham Links on page 35</a>

## multihop

<b>Syntax</b>	<pre>multihop {     no-nexthop-change;     ttl <i>ttl-value</i>; }</pre>
<b>Hierarchy Level</b>	<pre>[edit logical-systems <i>logical-system-name</i> protocols bgp], [edit logical-systems <i>logical-system-name</i> protocols bgp group <i>group-name</i>], [edit logical-systems <i>logical-system-name</i> protocols bgp group <i>group-name</i> neighbor <i>address</i>], [edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols   bgp], [edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols   bgp group <i>group-name</i>], [edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols   bgp group <i>group-name</i> neighbor <i>address</i>], [edit protocols bgp], [edit protocols bgp group <i>group-name</i>], [edit protocols bgp group <i>group-name</i> neighbor <i>address</i>], [edit routing-instances <i>routing-instance-name</i> protocols bgp], [edit routing-instances <i>routing-instance-name</i> protocols bgp group <i>group-name</i>], [edit routing-instances <i>routing-instance-name</i> protocols bgp group <i>group-name</i>   neighbor <i>address</i>]</pre>
<b>Release Information</b>	<p>Statement introduced before Junos OS Release 7.4.</p> <p>Statement introduced in Junos OS Release 9.0 for EX Series switches.</p> <p>Statement introduced in Junos OS Release 11.3 for the QFX Series.</p>
<b>Description</b>	<p>Configure an EBGp multihop session.</p> <p>For Layer 3 VPNs, you configure the EBGp multihop session between the PE and CE routing devices. This allows you to configure one or more routing devices between the PE and CE routing devices.</p> <p>An external confederation peer is a special case that allows unconnected third-party next hops. You do not need to configure multihop sessions explicitly in this particular case because multihop behavior is implied.</p> <p>If you have external BGP confederation peer-to-loopback addresses, you still need the multihop configuration.</p>
	<div>  <p><b>NOTE:</b> You cannot configure the <code>accept-remote-nexthop</code> statement at the same time.</p> </div>
<b>Default</b>	<p>If you omit this statement, all EBGp peers are assumed to be directly connected (that is, you are establishing a nonmultihop, or “regular,” BGP session), and the default time-to-live (TTL) value is 1.</p> <p>The remaining statements are explained separately.</p>

<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <i>Example: Configuring EBGp Multihop Sessions</i></li><li>• <a href="#">Configuring EBGp Multihop Sessions Between PE and CE Routers in Layer 3 VPNs on page 49</a></li><li>• <i>accept-remote-nexthop</i></li><li>• <i>no-nexthop-change</i></li><li>• <i>tll</i></li></ul>


---

## multipath (Routing Options)

---

<b>Syntax</b>	<pre>multipath {     vpn-unequal-cost equal-external-internal;     as-path-compare; }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options], [edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> routing-options rib <i>routing-table-name</i> ], [edit routing-instances <i>routing-instance-name</i> routing-options], [edit routing-instances <i>routing-instance-name</i> routing-options rib <i>routing-table-name</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. <b>equal-external-internal</b> option added in Junos OS Release 8.4. <b>as-path-compare</b> option introduced in Junos OS Release 10.1 Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	<p>Enable protocol-independent load balancing for Layer 3 VPNs. This allows the forwarding next hops for both the active route and alternative paths to be used for load balancing. For IPv6, you must configure the <b>multipath</b> statement at both the [edit routing-instances <i>routing-instance-name</i> routing-options] hierarchy level and the [edit routing-instances <i>routing-instance-name</i> routing-options rib <i>routing-table-name</i>] hierarchy level.</p> <p>Protocol-independent load balancing is applied to VPN routes that are equal up to their router identifiers (<b>router-id</b>) with regard to route selection.</p> <p>The options are explained separately.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Configuring Protocol-Independent Load Balancing in Layer 3 VPNs on page 95</a></li></ul>

## no-vrf-advertise

<b>Syntax</b>	no-vrf-advertise;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> ], [edit routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	<p>Prevent advertising VPN routes from a VRF routing instance to remote PE routers.</p> <p>Within a hub-and-spoke configuration, you can configure a PE router to not advertise VPN routes from the primary (hub) routing instance. Instead, these routes are advertised from the secondary (downstream) routing instance. You can do this without configuring routing table groups, by using the <b>no-vrf-advertise</b> statement.</p>
<div>  <p><b>NOTE:</b> This statement does not prevent the exportation of VPN routes to other VRF routing instances on the same router by configuring the [edit routing-options auto-export] statement.</p> </div>	
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <i>Configuring Policies for the VRF Table on PE Routers in VPNs</i></li> <li>• <i>Configuring Hub-and-Spoke VPN Topologies: One Interface on page 132</i></li> <li>• <i>Limiting Routes to Be Advertised by an MVPN VRF Instance</i></li> <li>• <i>vrf-advertise-selective</i></li> </ul>

## no-vrf-propagate-ttl

---

<b>Syntax</b>	no-vrf-propagate-ttl;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> ], [edit protocols routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 10.4.
<b>Description</b>	Disable normal time-to-live (TTL) decrementing in a VRF routing instance. You configure this statement once per routing instance, and it affects only RSVP-signaled or LDP-signaled LSPs in the routing instance. When this router acts as an ingress router for an LSP, it pushes an MPLS header with a TTL value of 255, regardless of the IP packet TTL. When the router acts as the penultimate router, it pops the MPLS header without writing the MPLS TTL into the IP packet.
<b>Default</b>	Normal TTL decrementing enabled; the TTL field value is decremented by 1 as the packet passes through each label-switched router in the LSP.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Disabling Normal TTL Decrementing in a VRF Routing Instance on page 227</a></li><li>• <i>Disabling Normal TTL Decrementing</i> in the <i>Junos OS MPLS Applications Library for Routing Devices</i></li></ul>

## protect core

<b>Syntax</b>	protect core;
<b>Hierarchy Level</b>	[edit routing-instances <i>routing-instance-name</i> routing-options]
<b>Release Information</b>	Statement introduced in Junos OS Release 13.2.
<b>Description</b>	In an MPLS VPN Core network, enable BGP Prefix-Independent Convergence (PIC) Edge to install a VPN route in the forwarding table for use as an alternate path when a provider edge (PE) router fails or you lose connectivity to a PE router. This alternate link is used until global convergence through the interior gateway protocol (IGP) occurs. Both OSPF with LDP and IS-IS with LDP are supported.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Configuring BGP PIC Edge for MPLS Layer 3 VPNs on page 99</a></li> <li>• <a href="#">Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 59</a></li> <li>• <a href="#">Introduction to Configuring Layer 3 VPNs on page 28</a></li> <li>• <a href="#">LDP Operation</a></li> <li>• <a href="#">Understanding Multiprotocol BGP</a></li> </ul>

## protection (Protocols BGP)

<b>Syntax</b>	protection;
<b>Hierarchy Level</b>	[edit routing-instances <i>instance-name</i> protocols bgp family inet unicast], [edit routing-instances <i>instance-name</i> protocols bgp family inet6 unicast], [edit routing-instances <i>instance-name</i> protocols bgp family inet labeled-unicast], [edit routing-instances <i>instance-name</i> protocols bgp family inet6 labeled-unicast]
<b>Description</b>	Configure the backup path to protect the active provider edge path in a Layer 3 VPN or a BGP labeled unicast path.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Example: Configuring Provider Edge Link Protection in Layer 3 VPNs on page 59</a></li> <li>• <a href="#">Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths on page 332</a></li> </ul>

## routing-instances (Class of Service)

---

<b>Syntax</b>	<pre>routing-instances <i>routing-instance-name</i> {   classifiers {     dscp (<i>classifier-name</i>   default);     dscp-ipv6 (<i>classifier-name</i>   default);     exp (<i>classifier-name</i>   default);   } }</pre>
<b>Hierarchy Level</b>	[edit class-of-service]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4.
<b>Description</b>	For routing instances with VRF table labels enabled, apply a custom MPLS EXP classifier or DSCP classifier to the routing instance. You can apply the default MPLS EXP classifier or one that is previously defined.
<b>Default</b>	If you do not include this statement, the default MPLS EXP classifier is applied to the routing instance. When no DSCP classifier is configured, the default MPLS EXP classifier is applied.
<b>Options</b>	<p><i>routing-instance-name</i>—Name of a routing instance.</p> <p><i>classifier-name</i>—Name of the behavior aggregate MPLS EXP classifier or DSCP classifier.</p>
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <i>Configuring Forwarding Classes</i></li><li>• <a href="#">Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs on page 58</a></li></ul>

## sham-link

<b>Syntax</b>	sham-link { local <i>address</i> ; }
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols ospf], [edit routing-instances <i>routing-instance-name</i> protocols ospf]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4.
<b>Description</b>	<p>Configure the local endpoint of a sham link.</p> <p>You can create an intra-area link or sham link between two provider edge (PE) routing devices so that the VPN backbone is preferred over the back-door link. A back-door link is a backup link that connects customer edge (CE) devices in case the VPN backbone is unavailable. When such a backup link is available and the CE devices are in the same OSPF area, the default behavior is to prefer this backup link over the VPN backbone. This is because the backup link is considered an intra-area link, while the VPN backbone is always considered an inter-area link. Intra-area links are always preferred over inter-area links.</p> <p>The sham link is an unnumbered point-to-point intra-area link between PE devices. When the VPN backbone has a sham intra-area link, this sham link can be preferred over the backup link if the sham link has a lower OSPF metric than the backup link.</p> <p>The sham link is advertised using Type 1 link-state advertisements (LSAs). Sham links are valid only for routing instances and OSPFv2.</p> <p>Each sham link is identified by the combination of a local endpoint address and a remote endpoint address.</p>
<b>Options</b>	<b>local <i>address</i></b> —The address for the local endpoint of the sham link.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Example: Configuring OSPFv2 Sham Links on page 36</a></li> <li>• <a href="#">sham-link-remote on page 552</a></li> </ul>

## sham-link-remote

<b>Syntax</b>	<pre>sham-link-remote address {     demand-circuit;     ipsec-sa name;     metric metric; }</pre>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> protocols ospf area <i>area-id</i> ], [edit routing-instances <i>routing-instance-name</i> protocols ospf area <i>area-id</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Support for <b>ipsec-sa</b> statement added in Junos OS Release 8.3.
<b>Description</b>	<p>Configure the remote endpoint of a sham link.</p> <p>You can create an intra-area link or sham link between two provider edge (PE) routing devices so that the VPN backbone is preferred over the back-door link. A back-door link is a backup link that connects customer edge (CE) devices in case the VPN backbone is unavailable. When such a backup link is available and the CE devices are in the same OSPF area, the default behavior is to prefer this backup link over the VPN backbone. This is because the backup link is considered an intra-area link, while the VPN backbone is always considered an inter-area link. Intra-area links are always preferred over inter-area links.</p> <p>The sham link is an unnumbered point-to-point intra-area link between PE devices. When the VPN backbone has a sham intra-area link, this sham link can be preferred over the backup link if the sham link has a lower OSPF metric than the backup link.</p> <p>The sham link is advertised using Type 1 link-state advertisements (LSAs). Sham links are valid only for routing instances and OSPFv2.</p> <p>Each sham link is identified by the combination of a local endpoint address and a remote endpoint address.</p>
<b>Options</b>	<p><b>address</b>—Address for the remote end point of the sham link.</p> <p>The remaining statements are explained separately.</p>
<b>Required Privilege Level</b>	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Example: Configuring OSPFv2 Sham Links on page 36</a></li> <li>• <a href="#">sham-link on page 551</a></li> </ul>

## source-class-usage

<b>Syntax</b>	source-class-usage { <i>direction</i> ; }
<b>Hierarchy Level</b>	[edit interfaces <i>interface-name</i> unit <i>logical-unit-number</i> family inet accounting], [edit logical-systems <i>logical-system-name</i> interfaces <i>interface-name</i> unit <i>logical-unit-number</i> family inet accounting], [edit routing-instances <i>routing-instance-name</i> vrf-table-label]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Support for the <b>vrf-table-label</b> statement added in Junos OS Release 9.3.
<b>Description</b>	Enable packet counters on an interface that count packets that arrive from specific prefixes on the provider core router and are destined for specific prefixes on the customer edge router.
<b>Options</b>	<p><i>direction</i> can be one of the following:</p> <p><b>input</b>—Configure at least one expected ingress point.</p> <p><b>output</b>—Configure at least one expected egress point.</p> <p><b>input output</b>—On a single interface, configure at least one expected ingress point and one expect egress point.</p>
<b>Required Privilege Level</b>	<p>interface—To view this statement in the configuration.</p> <p>interface-control—To add this statement to the configuration.</p>
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <i>Enabling Source Class and Destination Class Usage</i></li> <li>• <i>accounting</i></li> <li>• <i>destination-class-usage</i></li> <li>• <i>Junos OS Services Interfaces Library for Routing Devices</i></li> <li>• <a href="#">vrf-table-label on page 557</a></li> </ul>

## supplementary-blackout-timer (Host Fast Reroute)

<b>Syntax</b>	<code>supplementary-blackout-timer <i>minutes</i>;</code>
<b>Hierarchy Level</b>	<code>[edit logical-systems <i>logical-system-name</i> routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>],</code> <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code>
<b>Release Information</b>	Statement introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Override the global supplementary blackout timer for the specified host fast-reroute (HFRR) profile.</p> <p>When you configure HFRR, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices.</p> <p>There are two configuration statements (<code>global-arp-prefix-limit</code> and <code>arp-prefix-limit</code>) that set the ARP prefix limit, one at the global <code>[edit routing-options host-fast-reroute]</code> hierarchy level and the other at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, respectively. The global <code>global-arp-prefix-limit</code> statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The <code>arp-prefix-limit</code> statement overrides the <code>global-arp-prefix-limit</code> for that HFRR profile for that protected interface.</p> <p>Warning system log messages begin when the ARP routes in an HFRR profile reaches 80% of the configured limit. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.</p> <p>After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.</p> <p>There are global and per-HFRR CLI statements (<code>global-supplementary-blackout-timer</code> and <code>supplementary-blackout-timer</code>) to configure the supplementary blackout timer. The global value is at the <code>[edit routing-options host-fast-reroute]</code> hierarchy level and applies to all HFRR profiles on the routing device. The value for the routing-instance interface is at the <code>[edit routing-instances <i>instance-name</i> routing-options interface <i>interface-name</i>]</code> hierarchy level, and overrides the global value for that HFRR profile only.</p> <p>When the blackout timer expires, the HFRR profile is reactivated, and the Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.</p> <p>If an HFRR profile is in the deactivated state, a reevaluation of the ARP state is performed during every commit operation or whenever the routing process (rpd) is restarted with the <code>restart routing</code> command.</p>
<b>Default</b>	If you omit the <code>supplementary-blackout-timer</code> statement, the <code>global-supplementary-blackout-timer</code> takes effect for all HFRR profiles on the device. If

you omit both of these statements, the blackout timeout value is set to the ARP cache timeout value, which by default is 20 minutes and is configurable with the **aging-timer** statement at the **[edit system arp]** hierarchy level.

**Options** *minutes*—Number of minutes, in addition to the ARP cache timeout value, before an HFRR profile is reactivated after the ARP prefix limit is exceeded.  
**Range:** 1 through 15 minutes

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

**Related Documentation**

- [Example: Configuring Link Protection with Host Fast Reroute on page 242](#)

## vpn-unequal-cost

**Syntax** `vpn-unequal-cost {  
     equal-external-internal;  
 }`

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

**Release Information** Statement introduced before Junos OS Release 7.4. The **equal-external-internal** option was added for Junos OS Release 8.4.  
 Statement introduced in Junos OS Release 12.3 for ACX Series routers.

**Description** Apply protocol-independent load balancing to VPN routes that are equal until their interior gateway protocol (IGP) metrics with regard to route selection. If you do not configure the **vpn-unequal-cost** statement, protocol-independent load balancing is applied to VPN routes that are equal until their router identifiers with regard to route selection.

**Options** **equal-external-internal**—Specifies that both external and internal BGP paths can be selected for multipath.

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

**Related Documentation**

- [Configuring Load Balancing for Layer 3 VPNs on page 95](#)
- [Load Balancing and IP Header Filtering for Layer 3 VPNs on page 59](#)

## vrf-propagate-ttl

---

<b>Syntax</b>	vrf-propagate-ttl;
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> ], [edit routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced in Junos OS Release 10.4.
<b>Description</b>	Enable normal time-to-live (TTL) decrementing in a VRF routing instance. You configure this statement once per routing instance, and it affects only RSVP-signaled or LDP-signaled LSPs in the routing instance. When this router acts as an ingress router for an LSP, it pushes an MPLS copies the TTL value from the IP packet header. When the router acts as the penultimate router, it pops the MPLS header and writes the MPLS TTL into the IP packet.
<b>Default</b>	Normal TTL decrementing enabled; the TTL field value is decremented by 1 as the packet passes through each label-switched router in the LSP. This statement explicitly configures the default behavior for the VRF routing instance and is useful for overriding the <b>no-propagate-ttl</b> configured globally on the router at the <b>[edit protocols mpls]</b> hierarchy level.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"><li>• <a href="#">Example: Disabling Normal TTL Decrementing in a VRF Routing Instance on page 227</a></li><li>• <i>Disabling Normal TTL Decrementing in the Junos OS MPLS Applications Library for Routing Devices</i></li></ul>

## vrf-table-label

<b>Syntax</b>	<code>vrf-table-label {     source-class-usage; }</code>
<b>Hierarchy Level</b>	[edit logical-systems <i>logical-system-name</i> routing-instances <i>routing-instance-name</i> ], [edit routing-instances <i>routing-instance-name</i> ]
<b>Release Information</b>	Statement introduced before Junos OS Release 7.4. Support for the <b>source-class-usage</b> statement added in Junos OS Release 9.3. Statement introduced in Junos OS Release 11.1 for EX Series switches. Statement introduced in Junos OS Release 12.3 for ACX Series routers.
<b>Description</b>	<p>Map the inner label of a packet to a specific VPN routing and forwarding (VRF) instance. This allows the examination of the encapsulated IP header. The first lookup is done on the VPN label to determine which VRF instance to refer to, and the second lookup is done on the IP header to determine how to forward packets to the correct end hosts.</p> <p>When you include the <b>vrf-table-label</b> statement in the configuration of a VRF routing instance, a label-switched interface (LSI) logical interface label is created and mapped to the VRF routing table. Any routes in the VRF routing table are advertised with the LSI logical interface label allocated for the VRF routing table. When packets destined for the VRF routing instance arrive on a core-facing interface, they are treated as if the enclosed IP packet arrived on the LSI interface and are then forwarded and filtered based on the correct table.</p> <p>All routes in a VRF routing instance configured with this option are advertised with one label allocated per VRF.</p>
<b>Options</b>	The remaining statement is explained separately.
<b>Required Privilege Level</b>	routing—To view this statement in the configuration. routing-control—To add this statement to the configuration.
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Filtering Packets in Layer 3 VPNs Based on IP Headers on page 51</a></li> <li>• <a href="#">Configuring EXP-Based Traffic Classification for VPLS</a></li> <li>• <a href="#">Load Balancing and IP Header Filtering for Layer 3 VPNs on page 59</a></li> </ul>



## PART 3

# Administration

- [Layer 3 VPNs Reference on page 561](#)
- [Operational Commands on page 563](#)



## CHAPTER 8

# Layer 3 VPNs Reference

- [Supported Layer 3 VPN Standards on page 561](#)

### Supported Layer 3 VPN Standards

---

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

- RFC 2283, *Multiprotocol Extensions for BGP-4*
- RFC 2685, *Virtual Private Networks Identifier*
- RFC 2858, *Multiprotocol Extensions for BGP-4*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4379, *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*

The traceroute functionality is supported only on transit routers.

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

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

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

#### Related Documentation

- [Supported Carrier-of-Carriers and Interprovider VPN Standards](#)
- [Supported VPWS Standards](#)
- [Supported Layer 2 VPN Standard](#)

- *Supported Multicast VPN Standards*
- *Supported VPLS Standards*
- *Supported MPLS Standards*
- *Supported BGP Standards*
- *Accessing Standards Documents on the Internet*

## CHAPTER 9

# Operational Commands

- `ping mpls l3vpn`
- `show hfr profiles`

## ping mpls l3vpn

---

**Syntax** ping mpls l3vpn prefix *prefix-name*  
<*l3vpn-name*>  
<bottom-label-ttl>  
<count *count*>  
<destination *address*>  
<detail>  
<exp *forwarding-class*>  
<logical-system (all | *logical-system-name*)>  
<size *bytes*>  
<source *source-address*>  
<sweep>

**Release Information** Command introduced before Junos OS Release 7.4.  
Command introduced in Junos OS Release 9.0 for EX Series switches.  
The **size** and **sweep** options were introduced in Junos OS Release 9.6.

**Description** Check the operability of an MPLS Layer 3 virtual private network (VPN) connection.  
Press Ctrl+c to interrupt a **ping mpls l3vpn** command.

**Options** **bottom-label-ttl**—(Optional) Display the time-to-live value for the bottom label in the label stack.

**count *count***—(Optional) Number of ping requests to send. If **count** is not specified, five ping requests are sent. The range of values is 1 through **1,000,000**. The default value is **5**.

**destination *address***—(Optional) Specify an address other than the default (**127.0.0.1/32**) for the ping echo requests. The address can be anything within the **127/8** subnet.

**detail**—(Optional) Display detailed information about the echo requests sent and received.

**exp *forwarding-class***—(Optional) Value of the forwarding class for the MPLS ping packets.

***l3vpn-name***—(Optional) Layer 3 VPN name.

**logical-system (all | *logical-system-name*)**—(Optional) Perform this operation on all logical systems or on the specified logical system.

**prefix *prefix-name***—Ping to test whether a prefix is present in a provider edge (PE) router's or switch's VPN routing and forwarding (VRF) table, by means of a Layer 3 VPN destination prefix. This option does not test the connection between a PE router or switch and a customer edge (CE) router or switch.

**size *bytes***—(Optional) Size of the label-switched path (LSP) ping request packet (**96** through **65468** bytes). Packets are 4-byte aligned. For example, If you enter a size of 97, 98, 99, or 100, the router or switch uses a size value of 100 bytes. If you enter a packet size that is smaller than the minimum size, an error message is displayed reminding you of the 96-byte minimum.

**source *source-address***—(Optional) IP address of the outgoing interface. This address is sent in the IP source address field of the ping request. If this option is not specified, the default address is usually the loopback interface (lo.0).

**sweep**—(Optional) Automatically determine the size of the maximum transmission unit (MTU).

**Additional Information** You must configure MPLS at the **[edit protocols mpls]** hierarchy level on the egress PE router or switch (the router or switch receiving the MPLS echo packets) to ping a Layer 2 circuit.

In asymmetric MTU scenarios, the echo response might be dropped. For example, if the MTU from System A to System B is 1000 bytes, the MTU from System B to System A is 500 bytes, and the ping request packet size is 1000 bytes. The echo response is dropped because the PAD TLV is included in the echo response, making it too large.

If the Layer 3 VPN traffic transits a route reflector within the network, the **ping mpls l3vpn** command does not work.

**Required Privilege Level** network

**List of Sample Output** [ping mpls l3vpn on page 565](#)  
[ping mpls l3vpn detail on page 565](#)

**Output Fields** 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. When an echo reply is received with an error code, the packets are not counted in the received packets count, and are counted separately.

## Sample Output

### ping mpls l3vpn

```
user@host> ping mpls l3vpn vpn1 prefix 10.255.245.122/32
!!!!
--- 1sping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

### ping mpls l3vpn detail

```
user@host> ping mpls l3vpn vpn1 prefix 10.255.245.122/32 detail
Request for seq 1, to interface 68, labels <100128, 100112>
Reply for seq 1, return code: Egress-ok
Request for seq 2, to interface 68, labels <100128, 100112>
Reply for seq 2, return code: Egress-ok
Request for seq 3, to interface 68, labels <100128, 100112>
Reply for seq 3, return code: Egress-ok
Request for seq 4, to interface 68, labels <100128, 100112>
Reply for seq 4, return code: Egress-ok
Request for seq 5, to interface 68, labels <100128, 100112>
Reply for seq 5, return code: Egress-ok
```

```
--- lsping statistics ---  
5 packets transmitted, 5 packets received, 0% packet loss
```

## show hfr profiles

<b>Syntax</b>	show hfr profiles <brief  extensive>
<b>Release Information</b>	Command introduced in Junos OS Release 12.2.
<b>Description</b>	<p>Display host fast reroute (HFRR) profile information.</p> <p>HFRR adds a precomputed protection path into the Packet Forwarding Engine, such that if a link between a provider edge device and a server farm becomes unusable for forwarding, the Packet Forwarding Engine can use another path without having to wait for the router or the protocols to provide updated forwarding information.</p>
<b>Options</b>	<p><b>none</b>—Display information about HRFF profiles.</p> <p><b>brief   extensive</b>—(Optional) Display the specified level of output.</p>
<b>Required Privilege Level</b>	view
<b>Related Documentation</b>	<ul style="list-style-type: none"> <li>• <a href="#">Example: Configuring Link Protection with Host Fast Reroute on page 242</a></li> </ul>
<b>List of Sample Output</b>	<a href="#">show hfr profiles on page 568</a>
<b>Output Fields</b>	<a href="#">Table 11 on page 567</a> describes the output fields for the <b>show hfr profiles</b> command. Output fields are listed in the approximate order in which they appear.

**Table 11: show hfr profiles Output Fields**

Field Name	Field Description
HFRR pointer	
HFRR current state	Status of the HFRR profile: <b>HFRR_ACTIVE</b> , <b>HFRR_INACTIVE</b> , <b>HFRR_IFLH-NOT-CONF</b> , and so on.
HFRR Prefix limit blackout timer expiry (in secs)	Time interval between an HFRR profile becoming inactive on exceeding the ARP prefix limit, and the profile starting the <b>SYNC</b> process.
HFRR prefix limit hit count	Number of times that an HFRR profile becomes inactive on exceeding the ARP prefix limit.
HFRR protected IFL name	Interface configured for the HFRR feature.
HFRR protected IFL handle	
HFRR routing instance name	The routing instance in which the HFRR interface is configured.
HFRR routing instance handle	

Table 11: show hfrf profiles Output Fields (*continued*)

Field Name	Field Description
HFRR sync BG scheduled	
HFRR RTS filter on	
HFRR delete BG scheduled	
HFRR ARP prefix limit	Configured ARP prefix limit.
HFRR ARP supplementary blackout timeout (in mins)	Supplementary time-out value configured for profile to be inactive when it hits ARP prefix limit.
HFRR number of ARP routes learned	Number of ARP routes learned on the configured interface.
HFRR number of FRR routes created	Number of ARP routes created on the configured interface.

## Sample Output

### show hfrf profiles

```

user@host> show hfrf profiles
HFRR pointer: 0x9254000
HFRR current state: HFRR_ACTIVE
HFRR Prefix limit blackout timer expiry (in secs): 0
HFRR prefix limit hit count: 0
HFRR protected IFL name: ge-4/1/0.0
HFRR protected IFL handle: 0x9248738
HFRR routing instance name: test
HFRR routing instance handle: 0x9145740
HFRR sync BG scheduled: NO
HFRR RTS filter on: YES
HFRR delete BG scheduled: NO
HFRR ARP prefix limit: 0
HFRR ARP supplementary blackout timeout (in mins): 1
HFRR number of ARP routes learned: 4
HFRR number of FRR routes created: 2

```

## PART 4

# Troubleshooting

- [Troubleshooting Layer 3 VPNs on page 571](#)



# Troubleshooting Layer 3 VPNs

- [Diagnosing Common Problems on page 571](#)
- [Example: Troubleshooting Layer 3 VPNs on page 574](#)

## Diagnosing Common Problems

---

**Problem**    **Description:** To troubleshoot problems in the Layer 3 VPN configuration, start at one end of the VPN (the local customer edge [CE] router) and follow the routes to the other end of the VPN (the remote CE router).

**Solution**    The following troubleshooting steps should help you diagnose common problems:

1. If you configured a routing protocol between the local provider edge (PE) and CE routers, check that the peering and adjacency are fully operational. When you do this, be sure to specify the name of the routing instance. For example, to check OSPF adjacencies, enter the **show ospf neighbor instance *routing-instance-name*** command on the PE router.

If the peering and adjacency are not fully operational, check the routing protocol configuration on the CE router and check the routing protocol configuration for the associated VPN routing instance on the PE router.

2. Check that the local CE and PE routers can ping each other.

To check that the local CE router can ping the VPN interface on the local PE router, use a **ping** command in the following format, specifying the IP address or name of the PE router:

```
user@host> ping (ip-address | host-name)
```

To check that the local PE router can ping the CE router, use a **ping** command in the following format, specifying the IP address or name of the CE router, the name of the interface used for the VPN, and the source IP address (the local address) in outgoing echo request packets:

```
user@host> ping ip-address interface interface local echo-address
```

Often, the peering or adjacency between the local CE and local PE routers must come up before a **ping** command is successful. To check that a link is operational in a lab setting, remove the interface from the VPN routing and forwarding (VRF) by deleting the **interface** statement from the **[edit routing-instance *routing-instance-name*]** hierarchy

level and recommitting the configuration. Doing this removes the interface from the VPN. Then try the **ping** command again. If the command is successful, configure the interface back into the VPN and check the routing protocol configuration on the local CE and PE routers again.

3. On the local PE router, check that the routes from the local CE router are in the VRF table (*routing-instance-name.inet.0*):

```
user@host> show route table routing-instance-name.inet.0 <detail>
```

The following example shows the routing table entries. Here, the loopback address of the CE router is **10.255.14.155/32** and the routing protocol between the PE and CE routers is BGP. The entry looks like any ordinary BGP announcement.

```
10.255.14.155/32 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Nexthop: 192.168.197.141 via fe-1/0/0.0, selected
            State: <Active Ext>
            Peer AS:      1
            Age: 45:46
            Task: BGP_1.192.168.197.141+179
            Announcement bits (2): 0-BGP.0.0.0.0+179 1-KRT
            AS path: 1 I
            Localpref: 100
            Router ID: 10.255.14.155
```

If the routes from the local CE router are not present in the VRF routing table, check that the CE router is advertising routes to the PE router. If static routing is used between the CE and PE routers, make sure the proper static routes are configured.

4. On a remote PE router, check that the routes from the local CE router are present in the *bgp.l3vpn.0* routing table:

```
user@host> show route table bgp.l3vpn.0 extensive
```

```
10.255.14.175:3:10.255.14.155/32 (1 entry, 0 announced)
  *BGP      Preference: 170/-101
            Route Distinguisher: 10.255.14.175:3
            Source: 10.255.14.175
            Nexthop: 192.168.192.1 via fe-1/1/2.0, selected
            label-switched-path vpn07-vpn05
            Push 100004, Push 100005(top)
            State: <Active Int Ext>
            Local AS:      69 Peer AS:      69
            Age: 15:27      Metric2: 338
            Task: BGP_69.10.255.14.175+179
            AS path: 1 I
            Communities: target:69:100
            BGP next hop: 10.255.14.175
            Localpref: 100
            Router ID: 10.255.14.175
            Secondary tables: VPN-A.inet.0
```

The output of the **show route table bgp.l3vpn.0 extensive** command contains the following information specific to the VPN:

- In the prefix name (the first line of the output), the route distinguisher is added to the route prefix of the local CE router. Because the route distinguisher is unique within the Internet, the concatenation of the route distinguisher and IP prefix provides unique VPN-IP version 4 (IPv4) routing entries.
- The **Route Distinguisher** field lists the route distinguisher separately from the VPN-IPv4 address.
- The **label-switched-path** field shows the name of the label-switched path (LSP) used to carry the VPN traffic.
- The **Push** field shows both labels being carried in the VPN-IPv4 packet. The first label is the inner label, which is the VPN label that was assigned by the PE router. The second label is the outer label, which is an RSVP label.
- The **Communities** field lists the target community.
- The **Secondary tables** field lists other routing tables on this router into which this route has been installed.

If routes from the local CE router are not present in the `bgp.l3vpn.0` routing table on the remote PE router, do the following:

- Check the VRF import filter on the remote PE router, which is configured in the **vrf-import** statement. (On the local PE router, you check the VRF export filter, which is configured with the **vrf-export** statement.)
- Check that there is an operational LSP or an LDP path between the PE routers. To do this, check that the IBGP next-hop addresses are in the `inet.3` table.
- Check that the IBGP session between the PE routers is established and configured properly.
- Check for “hidden” routes, which usually means that routes were not labeled properly. To do this, use the **show route table bgp.l3vpn.0 hidden** command.
- Check that the inner label matches the inner VPN label that is assigned by the local PE router. To do this, use the **show route table mpls** command.

The following example shows the output of this command on the remote PE router. Here, the inner label is **100004**.

```
...
Push 100004, Push 10005 (top)
```

The following example shows the output of this command on the local PE router, which shows that the inner label of **100004** matches the inner label on the remote PE router:

```
...
100004          *[VPN/7] 06:56:25, metric 1
> to 192.168.197.141 via fe-1/0/0.0, Pop
```

5. On the remote PE router, check that the routes from the local CE router are present in the VRF table (*routing-instance-name.inet.0*):

```
user@host> show route table routing-instance-name.inet.0 detail
10.255.14.155/32 (1 entry, 1 announced)
    *BGP      Preference: 170/-101
```

```
Route Distinguisher: 10.255.14.175:3
Source: 10.255.14.175
Nexthop: 192.168.192.1 via fe-1/1/2.0, selected
Label-switched-path vpn07-vpn05
Push 100004, Push 100005(top)
State: <Secondary Active Int Ext>
Local AS: 69 Peer AS: 69
Age: 1:16:22 Metric2: 338
Task: BGP_69.10.255.14.175+179
Announcement bits (2): 1-KRT 2-VPN-A-RIP
AS path: 1 I
Communities: target:69:100
BGP next hop: 10.255.14.175
Localpref: 100
Router ID: 10.255.14.175
Primary Routing Table bgp.l3vpn.0
```

In this routing table, the route distinguisher is no longer prepended to the prefix. The last line, **Primary Routing Table**, lists the table from which this route was learned.

If the routes are not present in this routing table, but were present in the `bgp.l3vpn.0` routing table on the local CE router, the routes might have not passed the VRF import policy on the remote PE router.

If a VPN-IPv4 route matches no **vrf-import** policy, the route does not show up in the `bgp.l3vpn` table at all and hence is not present in the VRF table. If this occurs, it might indicate that on the PE router, you have configured another **vrf-import** statement on another VPN (with a common target), and the routes show up in the `bgp.l3vpn.0` table, but are imported into the wrong VPN.

6. On the remote CE router, check that the routes from the local CE router are present in the routing table (`inet.0`):

```
user@host> show route
```

If the routes are not present, check the routing protocol configuration between the remote PE and CE routers, and make sure that peers and adjacencies (or static routes) between the PE and CE routers are correct.

7. If you determine that routes originated from the local CE router are correct, check the routes originated from the remote CE router by repeating this procedure.

---

## Example: Troubleshooting Layer 3 VPNs

This example shows how to use the **ping** command to check the accessibility of various routers in a VPN topology, and how to use the **traceroute** command to check the path that packets travel between the VPN routers.

- [Requirements on page 575](#)
- [Overview on page 575](#)
- [Pinging the CE Router from Another CE Router on page 577](#)
- [Pinging the Remote PE and CE Routers from the Local CE Router on page 578](#)
- [Pinging a CE Router from a Multiaccess Interface on page 579](#)
- [Pinging the Directly Connected PE Routers from the CE Routers on page 580](#)

- [Pinging the Directly Connected CE Routers from the PE Routers on page 581](#)
- [Pinging the Remote CE Router from the Local PE Router on page 582](#)
- [Troubleshooting Inconsistently Advertised Routes from Gigabit Ethernet Interfaces on page 583](#)

## Requirements

This example uses the following hardware and software components:

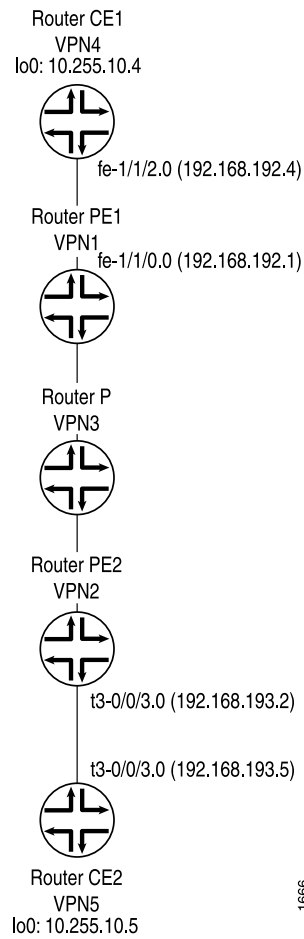
- M Series routers
- Junos OS Release 10.0R1 and later

## Overview

### Topology

The topology shown in [Figure 62 on page 576](#) illustrates the network used in this example to demonstrate how to employ the ping and traceroute commands to test connectivity between the routers participating in a Layer 3 VPN.

Figure 62: Layer 3 VPN Topology for ping and traceroute Examples



1666

## Pinging the CE Router from Another CE Router

**Step-by-Step Procedure** The following describes how to use the `ping` and `tracert` commands to troubleshoot Layer 3 VPN topologies. You can ping one CE router from the other by specifying the other CE router's loopback address as the IP address in the `ping` command. This `ping` command succeeds if the loopback addresses have been announced by the CE routers to their directly connected PE routers. The success of these `ping` commands also means that Router CE1 can ping any network devices beyond Router CE2, and vice versa. [Figure 62 on page 576](#) shows the topology referenced in the following steps:

1. Ping Router CE2 (VPN5) from Router CE1 (VPN4):

```
user@vpn4> ping 10.255.10.5 local 10.255.10.4 count 3
PING 10.255.10.5 (10.255.10.5): 56 data bytes
64 bytes from 10.255.10.5: icmp_seq=0 ttl=253 time=1.086 ms
64 bytes from 10.255.10.5: icmp_seq=1 ttl=253 time=0.998 ms
64 bytes from 10.255.10.5: icmp_seq=2 ttl=253 time=1.140 ms
--- 10.255.10.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.998/1.075/1.140/0.059 ms
```

2. To determine the path from Router CE1's loopback interface to Router CE2's loopback interface, use the `tracert` command:

```
user@vpn4> tracert 10.255.10.5 source 10.255.10.4
tracert to 10.255.10.5 (10.255.10.5) from 10.255.10.4, 30 hops max, 40 byte
packets
 1 vpn1-fe-110.isp-core.net (192.168.192.1) 0.680 ms 0.491 ms 0.456 ms
 2 vpn2-t3-001.isp-core.net (192.168.192.110) 0.857 ms 0.766 ms 0.754 ms

    MPLS Label=100005 CoS=0 TTL=1 S=1
 3 vpn5.isp-core.net (10.255.10.5) 0.825 ms 0.886 ms 0.732 ms
```

3. When you use the `tracert` command to examine the path used by a Layer 3 VPN, the provider (P) routers in the service provider's network are not displayed. As shown above, the jump from Router VPN1 to Router VPN2 is displayed as a single hop. The P router (VPN3) shown in [Figure 62 on page 576](#) is not displayed.

4. Ping Router CE1 (VPN4) from Router CE2 (VPN5):

```
user@vpn5> ping 10.255.10.4 local 10.255.10.5 count 3
PING 10.255.10.4 (10.255.10.4): 56 data bytes
64 bytes from 10.255.10.4: icmp_seq=0 ttl=253 time=1.042 ms
64 bytes from 10.255.10.4: icmp_seq=1 ttl=253 time=0.998 ms
64 bytes from 10.255.10.4: icmp_seq=2 ttl=253 time=0.954 ms
--- 10.255.10.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.954/0.998/1.042/0.036 ms
```

5. To determine the path from Router CE2 to Router CE1, use the `tracert` command:

```
user@vpn5> tracert 10.255.10.4 source 10.255.10.5
tracert to 10.255.10.4 (10.255.10.4) from 10.255.10.5, 30 hops max, 40 byte
packets
 1 vpn-08-t3-003.isp-core.net (192.168.193.2) 0.686 ms 0.519 ms 0.548 ms

 2 vpn1-so-100.isp-core.net (192.168.192.100) 0.918 ms 0.869 ms 0.859 ms

    MPLS Label=100021 CoS=0 TTL=1 S=1
 3 vpn4.isp-core.net (10.255.10.4) 0.878 ms 0.760 ms 0.739 ms
```

## Pinging the Remote PE and CE Routers from the Local CE Router

**Step-by-Step Procedure** From the local CE router, you can ping the VPN interfaces on the remote PE and CE routers, which are point-to-point interfaces. [Figure 62 on page 576](#) shows the topology referenced in the following examples:

1. Ping router CE2 from router CE1.

```
user@vpn4> ping 192.168.193.5 local 10.255.10.4 count 3
PING 192.168.193.5 (192.168.193.5): 56 data bytes
64 bytes from 192.168.193.5: icmp_seq=0 ttl=253 time=1.040 ms
64 bytes from 192.168.193.5: icmp_seq=1 ttl=253 time=0.891 ms
64 bytes from 192.168.193.5: icmp_seq=2 ttl=253 time=0.944 ms
--- 192.168.193.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.891/0.958/1.040/0.062 ms
```

2. To determine the path from Router CE1's loopback interface to Router CE2's directly connected interface, use the **traceroute** command:

```
user@vpn4> traceroute 192.168.193.5 source 10.255.10.4
traceroute to 192.168.193.5 (192.168.193.5) from 10.255.10.4, 30 hops max,
40 byte packets
 1 vpn1-fe-110.isp-core.net (192.168.192.1) 0.669 ms 0.508 ms 0.457 ms
 2 vpn2-t3-001.isp-core.net (192.168.192.110) 0.851 ms 0.769 ms 0.750 ms

    MPLS Label=100000 CoS=0 TTL=1 S=1
 3 vpn5-t3-003.isp-core.net (192.168.193.5) 0.829 ms 0.838 ms 0.731 ms
```

3. Ping Router PE2 (VPN2) from Router CE1 (VPN4). In this case, packets that originate at Router CE1 go to Router PE2, then to Router CE2, and back to Router PE2 before Router PE2 can respond to Internet Control Message Protocol (ICMP) requests. You can verify this by using the **traceroute** command.

```
user@vpn4> ping 192.168.193.2 local 10.255.10.4 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=254 time=1.080 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=254 time=0.967 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=254 time=0.983 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.967/1.010/1.080/0.050 ms
```

4. To determine the path from Router CE1 to Router PE2, use the **traceroute** command:

```
user@vpn4> traceroute 192.168.193.2 source 10.255.10.4
traceroute to 192.168.193.2 (192.168.193.2) from 10.255.10.4, 30 hops max,
40 byte packets
 1 vpn1-fe-110.isp-core.net (192.168.192.1) 0.690 ms 0.490 ms 0.458 ms
 2 vpn2-t3-003.isp-core.net (192.168.193.2) 0.846 ms 0.768 ms 0.749 ms

    MPLS Label=100000 CoS=0 TTL=1 S=1
 3 vpn5-t3-003.isp-core.net (192.168.193.5) 0.643 ms 0.703 ms 0.600 ms
 4 vpn-08-t3-003.isp-core.net (192.168.193.2) 0.810 ms 0.739 ms 0.729 ms
```

## Pinging a CE Router from a Multiaccess Interface

**Step-by-Step Procedure** You cannot ping one CE router from the other if the VPN interface is a multiaccess interface, such as the **fe-1/1/2.0** interface on Router CE1. To ping Router CE1 from Router CE2, you must either include the **vrf-table-label** statement at the **[edit routing-instances routing-instance-name]** hierarchy level on Router PE1 or configure a static route on Router PE1 to the VPN interface of Router CE1. If you include the **vrf-table-label** statement to ping a router, you cannot configure a static route.

1. If you configure a static route on Router PE1 to the VPN interface of Router CE1, its next hop must point to Router CE1 (at the **[edit routing-instance routing-instance-name]** hierarchy level), and this route must be announced from Router PE1 to Router PE2 as shown in the following configuration:

```
[edit]
routing-instances {
  direct-multipoint {
    instance-type vrf;
    interface fe-1/1/0.0;
    route-distinguisher 69:1;
    vrf-import direct-import;
    vrf-export direct-export;
    routing-options {
      static {
        route 192.168.192.4/32 next-hop 192.168.192.4;
      }
    }
  }
  protocols {
    bgp {
      group to-vpn4 {
        peer-as 1;
        neighbor 192.168.192.4;
      }
    }
  }
}
policy-options {
  policy-statement direct-export {
    term a {
      from protocol bgp;
      then {
        community add direct-comm;
        accept;
      }
    }
    term b {
      from {
        protocol static;
        route-filter 192.168.192.4/32 exact;
      }
      then {
        community add direct-comm;
        accept;
      }
    }
  }
}
```

```

    }
    term d {
        then reject;
    }
}
}
}

```

- Now you can ping Router CE1 from Router CE2:

```

user@vpn5> ping 192.168.192.4 local 10.255.10.5 count 3
PING 192.168.192.4 (192.168.192.4): 56 data bytes
64 bytes from 192.168.192.4: icmp_seq=0 ttl=253 time=1.092 ms
64 bytes from 192.168.192.4: icmp_seq=1 ttl=253 time=1.019 ms
64 bytes from 192.168.192.4: icmp_seq=2 ttl=253 time=1.031 ms
--- 192.168.192.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.019/1.047/1.092/0.032 ms

```

- To determine the path between these two interfaces, use the **traceroute** command:

```

user@vpn5> traceroute 192.168.192.4 source 10.255.10.5
traceroute to 192.168.192.4 (192.168.192.4) from 10.255.10.5, 30 hops max,
40 byte packets
 1  vpn-08-t3003.isp-core.net (192.168.193.2)  0.678 ms  0.549 ms  0.494 ms

 2  vpn1-so-100.isp-core.net (192.168.192.100)  0.873 ms  0.847 ms  0.844 ms

      MPLS Label=100021 CoS=0 TTL=1 S=1
 3  vpn4-fe-112.isp-core.net (192.168.192.4)  0.825 ms  0.743 ms  0.764 ms

```

## Pinging the Directly Connected PE Routers from the CE Routers

**Step-by-Step Procedure** From the loopback interfaces on the CE routers, you can ping the VPN interface on the directly connected PE router. [Figure 62 on page 576](#) shows the topology referenced in this procedure:

- From the loopback interface on Router CE1 (VPN4), ping the VPN interface, **fe-1/1/0.0**, on Router PE1:

```

user@vpn4> ping 192.168.192.1 local 10.255.10.4 count 3
PING 192.168.192.1 (192.168.192.1): 56 data bytes
64 bytes from 192.168.192.1: icmp_seq=0 ttl=255 time=0.885 ms
64 bytes from 192.168.192.1: icmp_seq=1 ttl=255 time=0.757 ms
64 bytes from 192.168.192.1: icmp_seq=2 ttl=255 time=0.734 ms
--- 192.168.192.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.734/0.792/0.885/0.066 ms

```

- From the loopback interface on Router CE2 (VPN5), ping the VPN interface, **t3-0/0/3.0**, on Router PE2:

```

user@vpn5> ping 192.168.193.2 local 10.255.10.5 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=255 time=0.998 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=255 time=0.834 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=255 time=0.819 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.819/0.884/0.998/0.081 ms

```

3. From the loopback interface on Router CE2 (VPN5), ping the VPN interface, **t3-0/0/3.0**, on Router PE2:

```
user@vpn5> ping 192.168.193.2 local 10.255.10.5 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=255 time=0.998 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=255 time=0.834 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=255 time=0.819 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.819/0.884/0.998/0.081 ms
```

4. To determine the path from the loopback interface on Router CE2 to the VPN interfaces on Router PE2, use the **traceroute** command:

```
user@vpn5> traceroute 192.168.193.2 source 10.255.10.5
traceroute to 192.168.193.2 (192.168.193.2) from 10.255.10.5, 30 hops max,
40 byte packets
 1 vpn-08-t3003.isp-core.net (192.168.193.2) 0.852 ms 0.670 ms 0.656 ms
```

## Pinging the Directly Connected CE Routers from the PE Routers

**Step-by-Step Procedure** From the VPN and loopback interfaces on the PE routers, you can ping the VPN interface on the directly connected CE router. [Figure 62 on page 576](#) shows the topology referenced in this procedure:

1. From the VPN interface on the PE router (router PE1), you can ping the VPN or loopback interface on the directly connected CE router (router CE1).

From the VPN interface on Router PE1 (VPN1), ping the VPN interface, **fe-1/1/0.0**, on Router CE1:

```
user@vpn1> ping 192.168.192.4 interface fe-1/1/0.0 local 192.168.192.1 count 3
PING 192.168.192.4 (192.168.192.4): 56 data bytes
64 bytes from 192.168.192.4: icmp_seq=0 ttl=255 time=0.866 ms
64 bytes from 192.168.192.4: icmp_seq=1 ttl=255 time=0.728 ms
64 bytes from 192.168.192.4: icmp_seq=2 ttl=255 time=0.753 ms
--- 192.168.192.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.728/0.782/0.866/0.060 ms
```

2. From the VPN interface on Router PE1 (VPN1), ping the loopback interface, **10.255.10.4**, on Router CE1:

```
user@vpn1> ping 10.255.10.4 interface fe-1/1/0.0 local 192.168.192.1 count 3
PING 10.255.10.4 (10.255.10.4): 56 data bytes
64 bytes from 10.255.10.4: icmp_seq=0 ttl=255 time=0.838 ms
64 bytes from 10.255.10.4: icmp_seq=1 ttl=255 time=0.760 ms
64 bytes from 10.255.10.4: icmp_seq=2 ttl=255 time=0.771 ms
--- 10.255.10.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.760/0.790/0.838/0.034 ms
```

3. To determine the path from the VPN interface on Router PE1 to the VPN and loopback interfaces on Router CE1, respectively, use the following **traceroute** commands:

```
user@vpn1> traceroute 10.255.10.4 interface fe-1/1/0.0 source 192.168.192.1
traceroute to 10.255.10.4 (10.255.10.4) from 192.168.192.1, 30 hops max,
40 byte packets
```

- ```

1  vpn4.isp-core.net (10.255.10.4)  0.842 ms  0.659 ms  0.621 ms
user@vpn1> traceroute 192.168.192.4 interface fe-1/1/0.0 source 192.168.192.1

traceroute to 192.168.192.4 (192.168.192.4) from 192.168.192.1, 30 hops max,
40 byte packets
1  vpn4-fe-112.isp-core.net (192.168.192.4)  0.810 ms  0.662 ms  0.640 ms

```
4. From the VPN interface on Router PE2 (VPN2), ping the VPN interface, **t3-0/0/3.0**, on Router CE2:

```

user@vpn2> ping 192.168.193.5 interface t3-0/0/3.0 local 192.168.193.2 count 3
PING 192.168.193.5 (192.168.193.5): 56 data bytes
64 bytes from 192.168.193.5: icmp_seq=0 ttl=255 time=0.852 ms
64 bytes from 192.168.193.5: icmp_seq=1 ttl=255 time=0.909 ms
64 bytes from 192.168.193.5: icmp_seq=2 ttl=255 time=0.793 ms
--- 192.168.193.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.793/0.851/0.909/0.047 ms

```
  5. From the VPN interface on Router PE2 (VPN2), ping the loopback interface, **10.255.10.5**, on Router CE2:

```

user@vpn2> ping 10.255.10.5 interface t3-0/0/3.0 local 192.168.193.2 count 3
PING 10.255.10.5 (10.255.10.5): 56 data bytes
64 bytes from 10.255.10.5: icmp_seq=0 ttl=255 time=0.914 ms
64 bytes from 10.255.10.5: icmp_seq=1 ttl=255 time=0.888 ms
64 bytes from 10.255.10.5: icmp_seq=2 ttl=255 time=1.066 ms
--- 10.255.10.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.888/0.956/1.066/0.079 ms

```
  6. To determine the path from the VPN interface on Router PE2 to the VPN and loopback interfaces on Router CE2, respectively, use the following **traceroute** commands:

```

user@vpn2> traceroute 10.255.10.5 interface t3-0/0/3.0 source 192.168.193.2
traceroute to 10.255.10.5 (10.255.10.5) from 192.168.193.2, 30 hops max,
40 byte packets
1  vpn5.isp-core.net (10.255.10.5)  1.009 ms  0.677 ms  0.633 ms
user@vpn2> traceroute 192.168.193.5 interface t3-0/0/3.0 source 192.168.193.2

traceroute to 192.168.193.5 (192.168.193.5) from 192.168.193.2, 30 hops max,
40 byte packets
1  vpn5-t3-003.isp-core.net (192.168.193.5)  0.974 ms  0.665 ms  0.619 ms

```

## Pinging the Remote CE Router from the Local PE Router

**Step-by-Step Procedure** The following procedure is effective for Layer 3 VPNs only. To ping a remote CE router from a local PE router in a Layer 3 VPN, you need to configure the following interfaces:

1. Configure a logical unit for the loopback interface.

To configure an additional logical unit on the loopback interface of the PE router, configure the **unit** statement at the **[edit interfaces lo0]** hierarchy level:

```

[edit interfaces]
lo0 {
  unit number {
    family inet {
      address address;

```

```

    }
  }
}

```

2. Configure the loopback interface for the Layer 3 VPN routing instance on the local PE router. You can associate one logical loopback interface with each Layer 3 VPN routing instance, enabling you to ping a specific routing instance on a router.

Specify the loopback interface you configured in Step 1 using the **interface** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

```

[edit routing-instances routing-instance-name]
  interface interface-name;

```

The ***interface-name*** is the logical unit on the loopback interface (for example, **lo0.1**).

3. From the VPN interface on PE router, you can now ping the logical unit on the loopback interface on the remote CE router:

```

user@host> ping interface interface host

```

Use ***interface*** to specify the new logical unit on the loopback interface (for example, **lo0.1**). For more information about how to use the **ping interface** command, see the *Junos Interfaces Command Reference*.

## Troubleshooting Inconsistently Advertised Routes from Gigabit Ethernet Interfaces

**Step-by-Step Procedure** For direct routes on a LAN in a Layer 3 VPN, the Junos OS attempts to locate a CE router that can be designated as the next hop. If this cannot be done, advertised routes from Gigabit Ethernet interfaces are dropped.

In such instances:

1. Use the **static** statement at the **[edit routing-options]** or **[edit logical-systems *logical-system-name* routing-options]** hierarchy levels in the VRF routing instance to a CE router on the LAN subnet, configuring the CE router as the next hop. All traffic to directly destinations on this LAN will go to the CE router. You can add two static routes to two CE routers on the LAN for redundancy.
2. Configure the **vrf-table-label** statement at the **[edit routing-instances *routing-instance-name*]** hierarchy levels to map the inner label of a packet to a specific VRF routing table. This allows the examination of the encapsulated IP header to force IP lookups on the VRF routing instance for all traffic.



**NOTE:** The **vrf-table-label** statement is not available for every core-facing interface; for example, channelized interfaces are not supported. See [“Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering” on page 54](#) for information about support for the **vrf-table-label** statement over Ethernet and SONET/SDH interfaces.



## PART 5

# Index

- [Index on page 587](#)



# Index

## Symbols

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

## A

|                                 |     |
|---------------------------------|-----|
| advertise-mode statement        |     |
| MPLS.....                       | 515 |
| arp-prefix-limit statement..... | 516 |
| as-override                     |     |
| usage guidelines.....           | 258 |
| as-path-compare                 |     |
| usage guidelines.....           | 97  |
| as-path-compare statement.....  | 517 |
| autonomous system number        |     |
| Layer 3 VPNs.....               | 31  |

## B

|                                          |     |
|------------------------------------------|-----|
| BGP                                      |     |
| import policy                            |     |
| family qualifier for.....                | 234 |
| Layer 3 VPNs.....                        | 31  |
| multihop sessions.....                   | 545 |
| routing to CE devices.....               | 258 |
| BGP labeled unicast                      |     |
| egress protection.....                   | 77  |
| BGP PIC Edge                             |     |
| configuring.....                         | 99  |
| example.....                             | 101 |
| BOOTP                                    |     |
| service.....                             | 85  |
| braces, in configuration statements..... | xx  |
| brackets                                 |     |
| angle, in syntax descriptions.....       | xx  |
| square, in configuration statements..... | xx  |

## C

|                                           |     |
|-------------------------------------------|-----|
| chained-composite-next-hop statement..... | 518 |
|-------------------------------------------|-----|

|                                                |     |
|------------------------------------------------|-----|
| classifiers statement.....                     | 519 |
| comments, in configuration statements.....     | xx  |
| composite next-hops.....                       | 112 |
| connections                                    |     |
| testing                                        |     |
| MPLS Layer 3 VPN connections.....              | 564 |
| conventions                                    |     |
| text and syntax.....                           | xix |
| curly braces, in configuration statements..... | xx  |
| customer support.....                          | xxi |
| contacting JTAC.....                           | xxi |

## D

|                                |     |
|--------------------------------|-----|
| demand-circuit statement       |     |
| usage guidelines.....          | 36  |
| description statement.....     | 519 |
| documentation                  |     |
| comments on.....               | xxi |
| domain-id statement.....       | 520 |
| domain-vpn-tag statement.....  | 520 |
| dynamic tunnels.....           | 521 |
| dynamic-tunnels statement..... | 521 |
| usage guidelines.....          | 89  |

## E

|                                |             |
|--------------------------------|-------------|
| EBGP                           |             |
| multihop for Layer 3 VPNs..... | 49          |
| egress filtering.....          | 51, 59      |
| egress protection              |             |
| BGP labeled unicast.....       | 77, 79, 345 |
| Layer 3 VPN.....               | 283         |
| egress-protection statement    |             |
| BGP.....                       | 524         |
| MPLS.....                      | 522, 523    |
| extended-space statement.....  | 525         |
| usage guidelines.....          | 113         |

## F

|                                      |          |
|--------------------------------------|----------|
| fast reroute                         |          |
| for hosts.....                       | 242, 247 |
| filtering packets, Layer 3 VPNs..... | 51       |
| font conventions.....                | xix      |
| forwarding-table statement.....      | 526      |

## G

|                                        |     |
|----------------------------------------|-----|
| global-arp-prefix-limit statement..... | 527 |
| global-supplementary-blackout-timer    |     |
| statement.....                         | 529 |

|                                                     |          |
|-----------------------------------------------------|----------|
| GRE tunnels                                         |          |
| configuration example.....                          | 198      |
| configuring dynamically.....                        | 89       |
| configuring manually.....                           | 87       |
| Layer 3 VPNs.....                                   | 86       |
| remote CE router.....                               | 89       |
| group-address statement.....                        | 531      |
| <b>H</b>                                            |          |
| host fast reroute.....                              | 242, 247 |
| host-fast-reroute statement.....                    | 532      |
| hosts, reachability                                 |          |
| MPLS Layer 3 VPN connections.....                   | 564      |
| hub-and-spoke, Layer 3 VPNs.....                    | 132, 144 |
| <b>I</b>                                            |          |
| IBGP                                                |          |
| Layer 3 VPNs.....                                   | 49       |
| multihop for Layer 3 VPNs.....                      | 49       |
| implementing interprovider layer 3 VPN option c     |          |
| configuring.....                                    | 444      |
| import statement                                    |          |
| route resolution                                    |          |
| usage guidelines.....                               | 238, 239 |
| independent domain                                  |          |
| overview.....                                       | 49       |
| independent-domain statement.....                   | 533      |
| example.....                                        | 358      |
| Layer 3 VPNs                                        |          |
| usage guidelines.....                               | 49       |
| usage guidelines.....                               | 49       |
| inet-vpn statement                                  |          |
| usage guidelines.....                               | 234      |
| inet6 statement.....                                | 535      |
| usage guidelines.....                               | 115      |
| inet6-vpn statement.....                            | 536      |
| usage guidelines.....                               | 234      |
| ingress statement.....                              | 534      |
| inter AS                                            |          |
| VPN, option B.....                                  | 82       |
| interface statement                                 |          |
| host fast reroute.....                              | 537      |
| interfaces descriptive text.....                    | 519      |
| Internet draft draft-marques-ppvnp-ibgp-version.txt |          |
| RFC 2547bis Networks Using Internal BGP as          |          |
| PE-CE Protocol.....                                 | 358      |
| interprovider layer 3 VPN option b                  |          |
| configuring.....                                    | 425      |
| interprovider Layer 3 VPN option C.....             | 444      |
| IP header filtering.....                            | 59, 213  |
| ipsec-sa statement                                  |          |
| OSPF                                                |          |
| usage guidelines.....                               | 36       |
| IPv4                                                |          |
| Layer 3 VPN load balancing.....                     | 95       |
| IPv6                                                |          |
| Layer 3 VPN load balancing.....                     | 95       |
| tunneling over Layer 3 VPN.....                     | 358      |
| <b>L</b>                                            |          |
| l3vpn statement.....                                | 538      |
| usage guidelines.....                               | 112      |
| label statement.....                                | 539      |
| usage guidelines.....                               | 81       |
| labeled-bgp statement.....                          | 540      |
| usage guidelines.....                               | 115      |
| labels                                              |          |
| allocation and substitution policy.....             | 81       |
| Layer 3 VPN                                         |          |
| egress protection.....                              | 283      |
| service mirroring.....                              | 283      |
| tunneling                                           |          |
| IPv6.....                                           | 358      |
| Layer 3 VPNs                                        |          |
| BGP PIC Edge                                        |          |
| configuring.....                                    | 99       |
| example.....                                        | 101      |
| composite next-hops.....                            | 112      |
| filtering packets.....                              | 51       |
| GRE tunnels, configuring.....                       | 86       |
| hub-and-spoke VPN topology                          |          |
| one interface.....                                  | 132      |
| two interfaces.....                                 | 144      |
| load balancing, IP header filtering.....            | 59, 213  |
| load balancing, IPv4 and IPv6.....                  | 95       |
| reachability, testing.....                          | 564      |
| route reflectors.....                               | 132      |
| Layer 3 VPNs                                        |          |
| GRE tunnels.....                                    | 198      |
| IBGP                                                |          |
| enabling transit of traffic.....                    | 49       |
| multihop EBGp and IBGP.....                         | 49       |
| OSPF                                                |          |
| configuring version 2.....                          | 32       |
| configuring version 3.....                          | 32       |
| site of origin.....                                 | 4        |
| target VPN.....                                     | 4        |
| VPN of origin.....                                  | 4        |

- 
- link-protection statement.....540
    - for host fast reroute
    - usage guidelines.....247
  - linking VRF tables between autonomous systems
    - configuring.....405
  - load balancing
    - Layer 3 VPNs.....59
    - Layer 3 VPNs, example.....213
  - local statement
    - OSPF.....551
    - usage guidelines.....36
  - LSPs
    - TTL decrementing, disabling in routing
      - instance.....548
    - TTL decrementing, enabling in routing
      - instance.....556
  - M**
  - manuals
    - comments on.....xxi
  - maximum-paths statement.....541
    - configuration guidelines.....44
  - maximum-prefixes statement.....543
    - configuration guidelines.....44
  - metric statement.....544
    - OSPF
      - usage guidelines.....36
  - MPLS
    - Layer 3 VPN connections
      - operability, checking.....564
  - multihop EBGp and IBGP.....49
  - multihop statement.....545
    - Layer 3 VPNs
      - usage guidelines.....49
  - multipath statement.....546
    - Layer 3 VPNs
      - usage guidelines.....95
  - N**
  - no-vrf-advertise statement.....547
  - no-vrf-propagate-ttl statement.....548
    - usage guidelines.....227
  - O**
  - OSPF
    - domain ID
      - example.....178
    - Layer 3 VPNs and IPv6.....48
    - sham link.....551, 552
  - OSPFv2
    - sham link.....35
  - P**
  - parentheses, in syntax descriptions.....xx
  - peer-as statement
    - usage guidelines
      - Layer 3 VPNs.....31
  - physical interfaces, descriptive text.....519
  - PIM
    - configuration statements.....531
  - ping mpls l3vpn command.....564
  - policies
    - label allocation and substitution.....81
  - protect core configuration statement.....549
  - R**
  - resolution statement
    - usage guidelines.....238, 239
  - resolution-ribs statement
    - usage guidelines.....238, 239
  - RFC 2547bis Networks Using Internal BGP as PE-CE
    - Protocol
      - Internet draft
        - draft-marques-ppvpn-ibgp-version.txt.....358
  - rib statement
    - route resolution
      - usage guidelines.....238, 239
  - RIP
    - routing instances, configure multiple.....33
  - route limit, configuring
    - paths.....541
    - prefix.....543
  - route reflector
    - in a Layer 3 VPN.....258
  - route reflectors
    - Layer 3 VPNs.....132
  - route resolution.....238, 239
  - route-filter statement
    - usage guidelines.....234
  - routing instances
    - RIP.....33
  - routing-instances statement.....550
    - usage guidelines.....58
  - S**
  - service mirroring
    - Layer 3 VPN.....283

|                                             |                       |
|---------------------------------------------|-----------------------|
| sham link                                   |                       |
| configuring.....                            | 36                    |
| overview.....                               | 35                    |
| sham-link statement.....                    | 551                   |
| usage guidelines.....                       | 36                    |
| sham-link-remote statement.....             | 552                   |
| usage guidelines.....                       | 36                    |
| signaled LSPs                               |                       |
| TTL decrementing in routing                 |                       |
| instance.....                               | 548, 556              |
| site of origin                              |                       |
| attribute of Layer 3 VPNs.....              | 4                     |
| source-class-usage statement.....           | 553                   |
| supplementary-blackout-timer statement..... | 554                   |
| support, technical                          | See technical support |
| syntax conventions.....                     | xix                   |

## T

|                                            |     |
|--------------------------------------------|-----|
| target VPN (attribute of Layer 3 VPN)..... | 4   |
| technical support                          |     |
| contacting JTAC.....                       | xxi |
| TTL                                        |     |
| disable decrementing in a VRF.....         | 227 |
| TTL decrementing                           |     |
| disabling in routing instance.....         | 548 |
| enabling in routing instance.....          | 556 |

## V

|                                                 |          |
|-------------------------------------------------|----------|
| verification                                    |          |
| host fast reroute (HFRR).....                   | 255      |
| static route.....                               | 239, 242 |
| VPN of origin (attribute of Layer 3 VPN).....   | 4        |
| vpn-unequal-cost statement.....                 | 555      |
| usage guidelines.....                           | 95       |
| VPNs                                            |          |
| label allocation and substitution policies..... | 81       |
| Layer 3                                         |          |
| supported software standards.....               | 561      |
| packet forwarding.....                          | 85       |
| VRF                                             |          |
| disable TTL decrementing.....                   | 227      |
| vrf-propagate-ttl statement.....                | 556      |
| vrf-table-label statement.....                  | 557      |