JUNIPER NETWORKS | Engineering Simplicity

Junos® OS

BGP User Guide

Published
2025-06-09

JUNOS

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

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, 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® OS BGP User Guide
Copyright © 2025 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 https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Guide | xxiv

1 Overview

BGP Overview | 2

Understanding BGP | 2

BGP Routes Overview | 6

BGP Route Resolution Overview | 7

BGP Messages Overview | 8

Understanding BGP RIB sharding and BGP Update IO thread | 11

Understanding BGP Path Selection | 12

Supported Standards for BGP | 17

2 Basic BGP Configurations

BGP Configuration Overview | 24

BGP Peering Sessions | 24

Advantages of Using External BGP Peer Groups | 25

Understanding External BGP Peering Sessions | 26

Example: Configuring External BGP Point-to-Point Peer Sessions | 27

Requirements | 28

Overview | 28

Configuration | 29

Verification | 33

Example: Configuring External BGP on Logical Systems with IPv6 Interfaces | 39

Requirements | 39

Overview | 40

Configuration | 42

Verification | 54

Understanding Internal BGP Peering Sessions | 61

Example: Configuring Internal BGP Peer Sessions | **63**

- Requirements | **63**
- Overview | **63**
- Configuration | **65**
- Verification | **76**

Example: Configuring Internal BGP Peering Sessions on Logical Systems | **80**

- Requirements | **80**
- Overview | **81**
- Configuration | **81**
- Verification | **90**

Overview: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same Link-Local Address (IPv6) | **94**

Example: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same IPv6 Link-Local Address | **95**

- Requirements | **95**
- Overview | **95**
- Configuration | **96**
- Verification | **100**

BGP Route Prioritization | 101

Understanding BGP Route Prioritization | **101**

Example: Configuring the BGP Output Priority Scheduler and Global Address Family Priority | **107**

- Requirements | **107**
- Overview | **107**
- Configuration | **108**
- Configure Global Output Priorities for a Route Family | **110**
- Configure a BGP Group Named test1 | **111**
- Verification | **112**

Example: Controlling Routing Table Convergence Using BGP Route Prioritization | **115**

- Requirements | **115**
- Overview | **116**
- Configure BGP Route Prioritization | **116**
- Verification | **119**

BGP Auto-Discovered Neighbors | 122

Understanding BGP Auto-discovered Neighbor | **123**

Example: Configuring BGP Auto-discovered Neighbor | **124**

- Overview | **125**
- Requirements | **126**
- Configuration | **126**
- Verification | **135**

3 **Configuring BGP Session Attributes**

Autonomous Systems for BGP Sessions | 141

Understanding the BGP Local AS Attribute | **141**

Example: Configuring a Local AS for EBGP Sessions | **146**

- Requirements | **147**
- Overview | **147**
- Configuration | **148**
- Verification | **157**

Example: Configuring a Private Local AS for EBGP Sessions | **162**

- Requirements | **162**
- Overview | **163**
- Configuration | **164**
- Verification | **169**

Understanding the Accumulated IGP Attribute for BGP | **171**

Example: Configuring the Accumulated IGP Attribute for BGP | **172**

- Requirements | **172**
- Overview | **172**
- Configuration | **174**
- Verification | **217**

Understanding AS Override | **227**

Example: Configuring a Layer 3 VPN with Route Reflection and AS Override | **228**

- Requirements | **228**
- Overview | **228**
- Configuration | **229**
- Verification | **240**

Example: Enabling BGP Route Advertisements | **242**

- Requirements | **243**
- Overview | **243**
- Configuration | **244**
- Verification | **251**

Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection | **254**

Example: Ignoring the AS Path Attribute When Selecting the Best Path | **255**

- Requirements | **255**
- Overview | **256**
- Configuration | **257**
- Verification | **264**

Understanding Private AS Number Removal from AS Paths | **265**

Example: Removing Private AS Numbers from AS Paths | **267**

- Requirements | **267**
- Overview | **267**
- Configuration | **268**
- Verification | **272**

Local Preference for BGP Routes | 276

Understanding Route Preference Values (Administrative Distance) | **276**

Example: Configuring the Preference Value for BGP Routes | **279**

- Requirements | **280**
- Overview | **280**
- Configuration | **282**
- Verification | **286**

Example: Using Routing Policy to Set a Preference Value for BGP Routes | **288**

- Requirements | **289**
- Overview | **289**
- Configuration | **290**
- Verification | **295**

Understanding the Local Preference Metric for Internal BGP Routes | **296**

Example: Configuring the Local Preference Value for BGP Routes | **297**

Requirements | **297**

Overview | **297**

Configuration | **298**

Verification | **313**

Example: Configuring BGP to Advertise Inactive Routes | **316**

Requirements | **317**

Overview | **317**

Configuration | **318**

Verification | **322**

BGP 4-Byte AS Numbers | 326

4-Byte Autonomous System Numbers Overview | **326**

Implementing 4-Byte Autonomous System Numbers | **327**

Configuring 4-Byte Autonomous System Numbers | **329**

Prepending 4-Byte AS Numbers in an AS Path | **330**

Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | **332**

Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | **333**

Understanding 4-Byte AS Numbers and Route Distinguishers | **337**

Understanding 4-Byte AS Numbers and Route Loop Detection | **338**

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | **340**

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | **341**

Example: Enforcing Correct Autonomous System Number in AS-Path in BGP Network | **343**

Requirements | **344**

Overview | **344**

Configure enforce-first-as Statement to Check Routes | **345**

Verification | **348**

BGP MED Attribute | 353

Understanding the MED Attribute That Determines the Exit Point in an AS | **353**

Example: Configuring the MED Attribute That Determines the Exit Point in an AS | **356**

Requirements | **357**
Overview | **357**
Configuration | **358**
Verification | **373**

Example: Configuring the MED Using Route Filters | **376**

Requirements | **376**
Overview | **376**
Configuration | **377**
Verification | **393**

Example: Configuring the MED Using Communities | **396**

Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates | **397**

Requirements | **397**
Overview | **397**
Configuration | **400**
Verification | **409**

BGP Multihop Sessions | 412

Understanding EBGP Multihop | **412**

Example: Configuring EBGP Multihop Sessions | **414**

Requirements | **414**
Overview | **414**
Configuration | **415**
Verification | **425**

4 **Configuring BGP Session Policies**

Basic BGP Routing Policies | 430

Understanding Routing Policies | **430**

Example: Applying Routing Policies at Different Levels of the BGP Hierarchy | **431**

Requirements | **431**
Overview | **431**
Configuration | **434**
Verification | **440**

Example: Injecting OSPF Routes into the BGP Routing Table | **444**

- Requirements | **444**
- Overview | **444**
- Configuration | **445**
- Verification | **448**
- Troubleshooting | **449**

Configuring Routing Policies to Control BGP Route Advertisements | **449**

Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers | **455**

- Requirements | **457**
- Overview | **457**
- Configuration | **458**
- Verification | **463**

Example: Configuring BGP Prefix-Based Outbound Route Filtering | **467**

- Requirements | **468**
- Overview | **468**
- Configuration | **469**
- Verification | **471**

Understanding the Default BGP Routing Policy on Packet Transport Routers (PTX Series) | **473**

Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers | **476**

- Requirements | **476**
- Overview | **476**
- Configuration | **477**
- Verification | **479**

Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases | **480**

Conditional Advertisement and Import Policy (Routing Table) with certain match conditions | **482**

Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table | **484**

- Requirements | **485**
- Overview | **485**
- Configuration | **489**
- Verification | **499**

Implicit filter for Default EBGP Route Propagation Behavior without Policies | **508**

Routing Policies for BGP Communities | 509

Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions | **510**

Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS | **512**

- Requirements | **512**
- Overview | **512**
- Configuration | **513**
- Verification | **524**

Example: Configuring a Routing Policy That Removes BGP Communities | **525**

- Requirements | **525**
- Overview | **526**
- Configuration | **527**
- Verification | **534**

Example: Configuring a Routing Policy Based on the Number of BGP Communities | **538**

- Requirements | **538**
- Overview | **538**
- Configuration | **539**
- Verification | **546**

5 **Enabling Load Balancing for BGP**

Load Balancing for a BGP Session | 550

Understanding BGP Multipath | **551**

Example: Load Balancing BGP Traffic | **552**

- Requirements | **552**
- Overview | **553**
- Configuration | **555**
- Verification | **558**

Understanding Configuration of Up to 512 Equal-Cost Paths With Optional Consistent Load Balancing | **561**

Example: Configuring Single-Hop EBGP Peers to Accept Remote Next Hops | **564**

- Requirements | **564**
- Overview | **564**

Configuration | **565**

Verification | **577**

Understanding Load Balancing for BGP Traffic with Unequal Bandwidth Allocated to the Paths | **581**

Example: Load Balancing BGP Traffic with Unequal Bandwidth Allocated to the Paths | **582**

Requirements | **582**

Overview | **583**

Configuration | **585**

Verification | **592**

Advertising Aggregate Bandwidth Across External BGP Links for Load Balancing Overview | **594**

Example: Configuring a Policy to Advertise Aggregate Bandwidth Across External BGP Links for Load Balancing | **596**

Requirements | **597**

Overview | **597**

Configuration | **598**

Verification | **607**

Understanding the Advertisement of Multiple Paths to a Single Destination in BGP | **610**

Example: Advertising Multiple Paths in BGP | **612**

Requirements | **613**

Overview | **613**

Configuration | **615**

Verification | **642**

Example: Configuring Selective Advertising of BGP Multiple Paths for Load Balancing | **649**

Requirements | **650**

Overview | **650**

Configuration | **651**

Verification | **661**

Example: Configuring a Routing Policy to Select and Advertise Multipaths Based on BGP Community Value | **667**

Requirements | **667**

Overview | **667**

Configuration | **669**

Verification | **677**

Configuring Recursive Resolution over BGP Multipath | **682**

Configuring ECMP Next Hops for RSVP and LDP LSPs for Load Balancing | **684**

Configuring Consistent Load Balancing for ECMP Groups | **686**

Improve Network Resiliency Using Multiple ECMP BGP Peers | **689**

- Overview | **689**
- Configuration | **690**
- Platform Support | **691**
- Related Documentation | **691**

Understanding Entropy Label for BGP Labeled Unicast LSP | **691**

Configuring an Entropy Label for a BGP Labeled Unicast LSP | **695**

Example: Configuring an Entropy Label for a BGP Labeled Unicast LSP | **697**

- Requirements | **697**
- Overview | **698**
- Configuration | **700**
- Verification | **714**

Use Case for BGP Prefix Independent Convergence for Inet, Inet6, or Labeled Unicast | **722**

Configuring BGP Prefix Independent Convergence for Inet | **723**

Example: Configuring BGP Prefix Independent Convergence for Inet | **728**

- Requirements | **729**
- Overview | **729**
- Configuration | **730**
- Verification | **743**

BGP PIC Edge Using BGP Labeled Unicast Overview | **748**

Configuring BGP PIC Edge Using BGP Labeled Unicast for Layer 2 Services | **750**

Example: Protecting IPv4 Traffic over Layer 3 VPN Running BGP Labeled Unicast | **751**

- Requirements | **751**
- Overview | **751**
- Configuration | **753**
- Verification | **819**

FAT Pseudowire Support for BGP L2VPN and VPLS Overview | **822**

Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **823**

Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **824**

- Requirements | **825**
- Overview | **825**
- Configuration | **826**
- Configuring PE2 | **844**
- Verification | **852**

Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **859**

Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **860**

- Requirements | **861**
- Overview | **861**
- Configuration | **862**
- Verification | **879**

BGP Egress Traffic Engineering | 882

Egress Peer Traffic Engineering Using BGP Labeled Unicast Overview | **882**

Configuring Egress Peer Traffic Engineering by Using BGP Labeled Unicast and Enabling MPLS Fast Reroute | **883**

Example: Configuring Egress Peer Traffic Engineering Using BGP Labeled Unicast | **886**

- Requirements | **887**
- Overview | **887**
- Configuration | **888**
- Verification | **907**

Segment Routing Traffic Engineering at BGP Ingress Peer Overview | **911**

Configuring Ingress Traffic Engineering with Segment Routing in a BGP Network | **915**

Enabling Traffic Statistics Collection for BGP Labeled Unicast | **919**

Understanding SRv6 Network Programming and Layer 3 Services over SRv6 in BGP | **921**

Example: Configuring Layer 3 Services over SRv6 in BGP Networks | **924**

- Requirements | **924**
- Overview | **924**
- Configuration | **926**
- Verification | **942**

Understanding SR-TE Policy for SRv6 Tunnel | **948**

Example: Configuring Static SR-TE Policy for an SRv6 Tunnel | **953**

Overview | **953**

Requirements | **954**

Configuration | **955**

Verification | **979**

Link-State Distribution Using BGP | 989

Link-State Distribution Using BGP Overview | **990**

Example: Configuring Link State Distribution Using BGP | **1005**

Requirements | **1005**

Overview | **1006**

Configuration | **1007**

Verification | **1022**

Configuring Link State Distribution Using BGP | **1029**

Link-State Distribution using SRv6 | **1033**

IPv6 Prefixes and IPv6 Adjacency SIDs MPLS Support in Traffic Engineering Database and BGP Link-State | **1036**

6 **Configuring Graceful Restart for BGP**

Understanding Graceful Restart for BGP | 1042

Understanding the Long-Lived BGP Graceful Restart Capability | **1042**

Understanding Maximum Period Configuration for Automatic Generation of BGP Keepalives by Kernel Timers After Switchover | **1044**

Interoperation of Functionalities With BGP Long-Lived Graceful Restart | **1045**

Monitoring and Administering BGP Long-Lived Graceful Restart | **1047**

Increasing the Duration for Preserving BGP Routes Across Slowly-Restarting Peers By BGP Long-Lived Graceful Restart | **1050**

Configuring BGP Long-Lived Graceful Restart Communities in Routing Policies | **1054**

Configuring Long-Lived Graceful Restarter Mode Negotiation for a Specific Address Family in Logical Systems and Routing Instances | **1057**

Informing the BGP Helper Router or Peer About Retaining Routes By Configuring the Forwarding State Bit for All Address Families and for a Specific Address Family | **1062**

Example: Preserving Route Details for Slow and Latent BGP Peers By Using BGP Long-Lived Graceful Restart | **1068**

- Requirements | **1068**
- Overview | **1069**
- Configuration | **1070**
- Verification | **1073**

7 **Configuring Multiprotocol for a BGP Session**

Multiprotocol BGP | 1076

Understanding Multiprotocol BGP | **1076**

Example: Configuring IPv6 BGP Routes over IPv4 Transport | **1084**

- Requirements | **1085**
- Overview | **1085**
- Configuration | **1086**
- Verification | **1091**

Advertising IPv4 Routes over BGP IPv6 Sessions Overview | **1094**

Example: Advertising IPv4 Routes over IPv6 BGP Sessions | **1095**

- Requirements | **1096**
- Overview | **1096**
- Configuration | **1097**
- Verification | **1102**

Understanding Redistribution of IPv4 Routes with IPv6 Next Hop into BGP | **1104**

Configuring BGP to Redistribute IPv4 Routes with IPv6 Next-Hop Addresses | **1111**

Enabling Layer 2 VPN and VPLS Signaling | **1114**

Understanding BGP Flow Routes for Traffic Filtering | **1115**

Example: Enabling BGP to Carry Flow-Specification Routes | **1123**

- Requirements | **1123**
- Overview | **1123**
- Configuration | **1126**
- Verification | **1137**

Example: Configuring BGP to Carry IPv6 Flow Specification Routes | **1145**

- Requirements | **1145**

Overview | **1145**

Configuration | **1146**

Verification | **1152**

Configuring BGP Flow Specification Action Redirect to IP to Filter DDoS Traffic | **1158**

Forwarding Traffic Using BGP Flow Specification DSCP Action | **1162**

Configuring Policies for Flow Route Validation | **1163**

8 **Configuring BGP CLNS**

BGP Connectionless Network Service (CLNS) | 1167

Understanding BGP for CLNS VPNs | **1167**

Enabling BGP to Carry CLNS Routes | **1168**

Example: Configuring BGP for CLNS VPNs | **1174**

Requirements | **1174**

Overview | **1174**

Configuration | **1174**

Verification | **1176**

9 **Using Route Reflectors and Confederations for BGP Networks**

BGP Route Reflectors | 1179

Understanding BGP Route Reflectors | **1179**

Example: Configuring a Route Reflector | **1182**

Requirements | **1183**

Overview | **1183**

Configuration | **1184**

Verification | **1197**

Understanding a Route Reflector That Belongs to Two Different Clusters | **1205**

Example: Configuring a Route Reflector That Belongs to Two Different Clusters | **1206**

Requirements | **1207**

Overview | **1207**

Configuration | **1207**

Verification | **1211**

Understanding BGP Optimal Route Reflection | **1213**

Configuring BGP Optimal Route Reflection on a Route Reflector to Advertise the Best Path | **1215**

BGP Route Server Overview | **1216**

BGP Confederations for IBGP Scaling | 1221

Understanding BGP Confederations | **1221**

Example: Configuring BGP Confederations | **1223**

- Requirements | **1223**
- Overview | **1223**
- Configuration | **1224**
- Verification | **1227**

10 **Configuring BGP Security**

BGP Route Authentication | 1234

Understanding Router Authentication for BGP | **1234**

TCP Authentication | **1235**

Example: Configuring Router Authentication for BGP | **1236**

- Requirements | **1236**
- Overview | **1237**
- Configuration | **1238**
- Verification | **1241**

IP Security for BGP | 1245

Understanding IPsec for BGP | **1245**

Example: Using IPsec to Protect BGP Traffic | **1246**

- Requirements | **1246**
- Overview | **1247**
- Configuration | **1248**
- Verification | **1250**

TCP Access Restriction for BGP | 1251

Understanding Security Options for BGP with TCP | **1252**

Example: Configuring a Filter to Block TCP Access to a Port Except from Specified BGP Peers | **1252**

- Requirements | **1253**
- Overview | **1253**
- Configuration | **1254**

Verification | **1258**

Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List | **1261**

Requirements | **1261**

Overview | **1261**

Configuration | **1262**

Verification | **1265**

Example: Limiting TCP Segment Size for BGP | **1266**

Requirements | **1266**

Overview | **1267**

Configuration | **1268**

Verification | **1271**

Troubleshooting | **1271**

BGP Origin Validation | 1274

Understanding Origin Validation for BGP | **1274**

Use Case and Benefit of Origin Validation for BGP | **1282**

Example: Configuring Origin Validation for BGP | **1283**

Requirements | **1284**

Overview | **1284**

Configuration | **1286**

Verification | **1299**

11 **Configuring BGP Route Flap Damping and Error Handling**

BGP Session and Route Flaps | 1306

Understanding BGP Session Resets | **1306**

Example: Preventing BGP Session Flaps When VPN Families Are Configured | **1307**

Requirements | **1307**

Overview | **1307**

Configuration | **1310**

Verification | **1315**

Understanding Damping Parameters | **1316**

Example: Configuring BGP Route Flap Damping Parameters | **1318**

Requirements | **1318**

Overview | **1318**

Configuration | **1319**

Verification | **1325**

Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family | **1332**

Requirements | **1332**

Overview | **1332**

Configuration | **1333**

Verification | **1345**

Understanding BGP-Static Routes for Preventing Route Flaps | **1347**

Configuring BGP-Static Routes for Preventing Route Flaps | **1348**

Example: Configuring BGP-Static Routes to Prevent Route Flaps | **1349**

Requirements | **1350**

Overview | **1350**

Configuration | **1352**

Verification | **1360**

BGP Error Messages | 1370

Understanding Error Handling for BGP Update Messages | **1370**

Example: Configuring Error Handling for BGP Update Messages | **1373**

Requirements | **1374**

Overview | **1374**

Configuration | **1376**

Verification | **1382**

BFD for BGP Sessions | 1387

Understanding BFD for BGP | **1387**

Example: Configuring BFD on Internal BGP Peer Sessions | **1388**

Requirements | **1388**

Overview | **1388**

Configuration | **1390**

Verification | **1396**

Understanding BFD Authentication for BGP | **1401**

Example: Configuring BFD Authentication for BGP | **1403**

Configuring BFD Authentication Parameters | **1403**

Viewing Authentication Information for BFD Sessions | **1405**

Platform-Specific BFD for BGP Behavior | **1407**

12 **Configuring BGP-Based VPN**

BGP-Based VPN | 1410

Understanding Carrier-of-Carriers VPNs | **1410**

Understanding Interprovider and Carrier-of-Carriers VPNs | **1412**

Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service | **1413**

BGP accept-own Community | 1423

Understanding BGP accept-own Community Attribute | **1423**

Configure BGP accept-own Community | **1425**

13 **Monitoring and Troubleshooting**

BGP Monitoring Protocol | 1428

Monitoring BGP Routing Information | **1428**

Understanding the BGP Monitoring Protocol | **1429**

Configuring BGP Monitoring Protocol Version 3 | **1430**

Configuring BGP Monitoring Protocol to Run Over a Different Routing Instance | **1431**

Configuring a Nondefault Routing Instance for BMP | **1431**

Configuring mgmt_junos for BMP | **1432**

Example: Configuring the BGP Monitoring Protocol | **1434**

Requirements | **1434**

Overview | **1434**

Configuration | **1435**

Verification | **1437**

Understanding Trace Operations for BGP Protocol Traffic | **1437**

Example: Viewing BGP Trace Files on Logical Systems | **1440**

Requirements | **1440**

Overview | **1440**

Configuration | **1441**

Verification | **1447**

Example: Tracing Global Routing Protocol Operations | **1447**

Requirements | **1448**

Overview | **1448**

Configuration | **1449**

Verification | **1453**

Tracing BMP Operations | **1454**

Troubleshooting Network Issues | 1456

Working with Problems on Your Network | **1456**

Isolating a Broken Network Connection | **1457**

Identifying the Symptoms of a Broken Network Connection | **1459**

Isolating the Causes of a Network Problem | **1461**

Taking Appropriate Action for Resolving the Network Problem | **1462**

Evaluating the Solution to Check Whether the Network Problem Is Resolved | **1464**

Checklist for Tracking Error Conditions | **1466**

Configure Routing Protocol Process Tracing | **1468**

Configure Routing Protocol Tracing for a Specific Routing Protocol | **1471**

Monitor Trace File Messages Written in Near-Real Time | **1474**

Stop Trace File Monitoring | **1475**

Troubleshooting BGP Sessions | 1476

Checklist for Verifying the BGP Protocol and Peers | **1477**

Verify BGP Peers | **1479**

Verify BGP on an Internal Router | **1481**

Verify BGP on a Border Router | **1485**

Verify Advertised BGP Routes | **1490**

Verify That a Particular BGP Route Is Received on Your Router | **1491**

Examine BGP Routes and Route Selection | **1492**

Examine the Local Preference Selection | **1495**

Examine the Multiple Exit Discriminator Route Selection | **1496**

Examine the EBGP over IBGP Selection | **1498**

Examine the IGP Cost Selection | **1500**

Checklist for Checking the BGP Layer | **1502**

Checking the BGP Layer | **1503**

Check That BGP Traffic Is Using the LSP | **1505**

Check BGP Sessions | **1506**

Verify the BGP Configuration | **1509**

Examine BGP Routes | **1516**

Verify Received BGP Routes | **1518**

Taking Appropriate Action for Resolving the Network Problem | **1520**

Check That BGP Traffic Is Using the LSP Again | **1521**

Display Sent or Received BGP Packets | **1523**

Understanding Hidden Routes | **1525**

Examine Routes in the Forwarding Table | **1527**

Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers | **1528**

Requirements | **1529**

Overview | **1529**

Configuration | **1530**

Verification | **1532**

Log BGP State Transition Events | **1533**

Configure BGP-Specific Options | **1536**

Display Detailed BGP Protocol Information | **1537**

Diagnose BGP Session Establishment Problems | **1539**

Configure IS-IS-Specific Options | **1541**

Displaying Detailed IS-IS Protocol Information | **1542**

Displaying Sent or Received IS-IS Protocol Packets | **1545**

Analyzing IS-IS Link-State PDUs in Detail | **1547**

Configure OSPF-Specific Options | **1550**

Diagnose OSPF Session Establishment Problems | **1551**

Analyze OSPF Link-State Advertisement Packets in Detail | **1556**

14 **Configuration Statements and Operational Commands**

peer-auto-discovery | 1560

Junos CLI Reference Overview | 1561

About This Guide

BGP is an exterior gateway protocol (EGP) that is used to exchange routing information among routers in different autonomous systems. The topics on this page provide information about BGP for devices running Junos OS.

1
CHAPTER

Overview

IN THIS CHAPTER

- BGP Overview | 2

BGP Overview

IN THIS SECTION

- Understanding BGP | 2
- BGP Routes Overview | 6
- BGP Route Resolution Overview | 7
- BGP Messages Overview | 8
- Understanding BGP RIB sharding and BGP Update IO thread | 11
- Understanding BGP Path Selection | 12
- Supported Standards for BGP | 17

Understanding BGP

IN THIS SECTION

- Autonomous Systems | 3
- AS Paths and Attributes | 3
- External and Internal BGP | 3
- Multiple Instances of BGP | 4
- Allow Protocol Traffic for Interfaces in a Security Zone | 5

BGP is an exterior gateway protocol (EGP) that is used to exchange routing information among routers in different autonomous systems (ASs). BGP routing information includes the complete route to each destination. BGP uses the routing information to maintain a database of network reachability information, which it exchanges with other BGP systems. BGP uses the network reachability information to construct a graph of AS connectivity, which enables BGP to remove routing loops and enforce policy decisions at the AS level.

Multiprotocol BGP (MBGP) extensions enable BGP to support IP version 6 (IPv6). MBGP defines the attributes MP_REACH_NLRI and MP_UNREACH_NLRI, which are used to carry IPv6 reachability

information. Network layer reachability information (NLRI) update messages carry IPv6 address prefixes of feasible routes.

BGP allows for policy-based routing. You can use routing policies to choose among multiple paths to a destination and to control the redistribution of routing information.

BGP uses TCP as its transport protocol, using port 179 for establishing connections. Running over a reliable transport protocol eliminates the need for BGP to implement update fragmentation, retransmission, acknowledgment, and sequencing.

The Junos OS routing protocol software supports BGP version 4. This version of BGP adds support for Classless Interdomain Routing (CIDR), which eliminates the concept of network classes. Instead of assuming which bits of an address represent the network by looking at the first octet, CIDR allows you to explicitly specify the number of bits in the network address, thus providing a means to decrease the size of the routing tables. BGP version 4 also supports aggregation of routes, including the aggregation of AS paths.

This section discusses the following topics:

Autonomous Systems

An *autonomous system* (AS) is a set of routers that are under a single technical administration and normally use a single interior gateway protocol and a common set of metrics to propagate routing information within the set of routers. To other ASs, an AS appears to have a single, coherent interior routing plan and presents a consistent picture of what destinations are reachable through it.

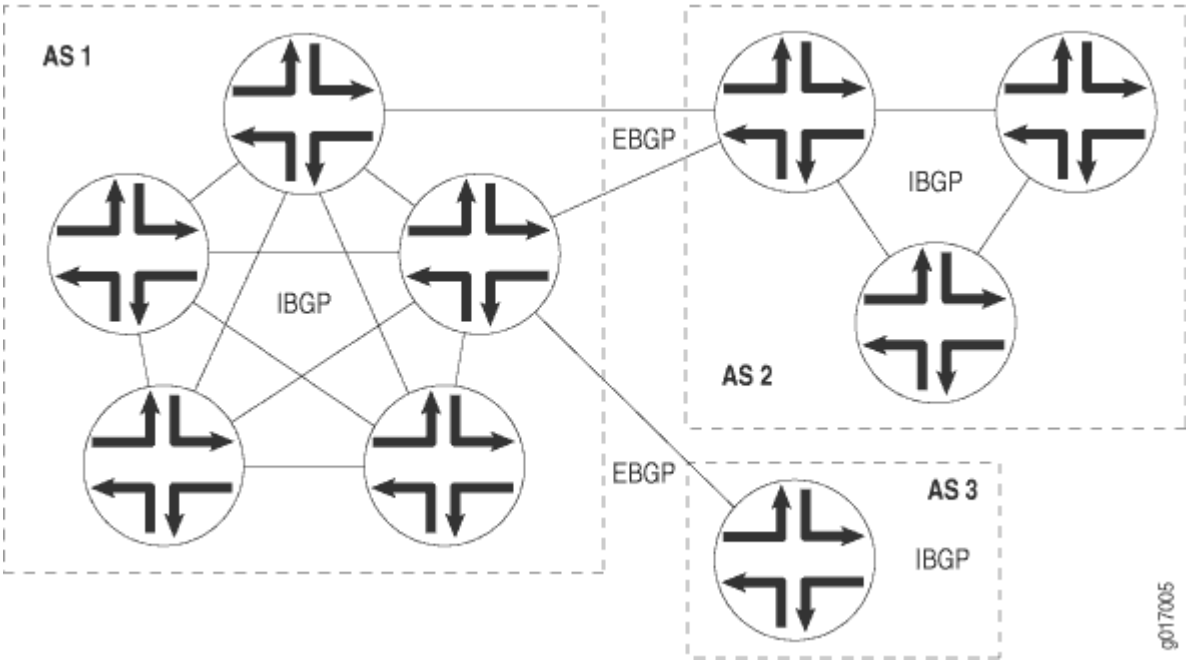
AS Paths and Attributes

The routing information that BGP systems exchange includes the complete route to each destination, as well as additional information about the route. The *AS path* is the sequence of autonomous systems the route traversed, and additional route information is included in *path attributes*. BGP uses the AS path and the path attributes to completely determine the network topology. Once BGP understands the topology, it can detect and eliminate routing loops and select among groups of routes to enforce administrative preferences and routing policy decisions.

External and Internal BGP

BGP supports two types of exchanges of routing information: exchanges among different ASs and exchanges within a single AS. When used among ASs, BGP is called *external BGP* (EBGP) and BGP sessions perform *inter-AS routing*. When used within an AS, BGP is called *internal BGP* (IBGP) and BGP sessions perform *intra-AS routing*. Figure 1 on page 4 illustrates ASs, IBGP, and EBGP.

Figure 1: ASs, EBGP, and IBGP



A BGP system shares network reachability information with adjacent BGP systems, which are referred to as *neighbors* or *peers*.

BGP systems are arranged into *groups*. In an IBGP group, all peers in the group—called *internal peers*—are in the same AS. Internal peers can be anywhere in the local AS and do not have to be directly connected to one another. Internal groups use routes from an IGP to resolve forwarding addresses. They also propagate external routes among all other internal routers running IBGP, computing the next hop by taking the BGP next hop received with the route and resolving it using information from one of the interior gateway protocols.

In an EBGP group, the peers in the group—called *external peers*—are in different ASs and normally share a subnet. In an external group, the next hop is computed with respect to the interface that is shared between the external peer and the local router.

Multiple Instances of BGP

You can configure multiple instances of BGP at the following hierarchy levels:

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

Multiple instances of BGP are primarily used for Layer 3 VPN support.

IGP peers and external BGP (EBGP) peers (both nonmultihop and multihop) are all supported for routing instances. BGP peering is established over one of the interfaces configured under the **routing-instances** hierarchy.

NOTE: When a BGP neighbor sends BGP messages to the local routing device, the incoming interface on which these messages are received must be configured in the same routing instance that the BGP neighbor configuration exists in. This is true for neighbors that are a single hop away or multiple hops away.

Routes learned from the BGP peer are added to the **instance-name.inet.0** table by default. You can configure import and export policies to control the flow of information into and out of the instance routing table.

For Layer 3 VPN support, configure BGP on the provider edge (PE) router to receive routes from the customer edge (CE) router and to send the instances' routes to the CE router if necessary. You can use multiple instances of BGP to maintain separate per-site forwarding tables for keeping VPN traffic separate on the PE router.

You can configure import and export policies that allow the service provider to control and rate-limit traffic to and from the customer.

You can configure an EBGP multihop session for a VRF routing instance. Also, you can set up the EBGP peer between the PE and CE routers by using the loopback address of the CE router instead of the interface addresses.

Allow Protocol Traffic for Interfaces in a Security Zone

On SRX Series Firewalls, you must enable the expected host-inbound traffic on the specified interfaces or all interfaces of the zone. Otherwise inbound traffic destined to this device is dropped by default.

For example, to allow BGP traffic on a specific zone of your SRX Series Firewall, use the following step:

```
[edit]
user@host# set security zones security-zone trust host-inbound-traffic protocols bgp
```

(All interfaces)

```
[edit]
user@host# set security zones security-zone trust interfaces ge-0/0/1.0 host-inbound-traffic
protocols bgp
```

(Specified interface)

SEE ALSO

BGP Messages Overview | 8

BGP Routes Overview

A BGP route is a destination, described as an IP address prefix, and information that describes the path to the destination.

The following information describes the path:

- AS path, which is a list of numbers of the ASs that a route passes through to reach the local router. The first number in the path is that of the last AS in the path—the AS closest to the local router. The last number in the path is the AS farthest from the local router, which is generally the origin of the path.
- Path attributes, which contain additional information about the AS path that is used in routing policy.

BGP peers advertise routes to each other in update messages.

BGP stores its routes in the Junos OS routing table (**inet.0**). The routing table stores the following information about BGP routes:

- Routing information learned from update messages received from peers
- Local routing information that BGP applies to routes because of local policies
- Information that BGP advertises to BGP peers in update messages

For each prefix in the routing table, the routing protocol process selects a single best path, called the active path. Unless you configure BGP to advertise multiple paths to the same destination, BGP advertises only the active path.

The BGP router that first advertises a route assigns it one of the following values to identify its origin. During route selection, the lowest origin value is preferred.

- **0**—The router originally learned the route through an IGP (OSPF, IS-IS, or a static route).
- **1**—The router originally learned the route through an EGP (most likely BGP).
- **2**—The route's origin is unknown.

SEE ALSO

Understanding BGP Path Selection | 12

Example: Advertising Multiple Paths in BGP | 612

BGP Route Resolution Overview

An internal BGP (IBGP) route with a next-hop address to a remote BGP neighbor (protocol next hop) must have its next hop resolved using some other route. BGP adds this route to the rpd resolver module for next-hop resolution. If RSVP is used in the network, then the BGP next hop is resolved using the RSVP ingress route. This results in the BGP route pointing to an indirect next hop, and the indirect next hop pointing to a forwarding next hop. The forwarding next hop is derived from the RSVP route next hop. There is often a large set of internal BGP routes that have the same protocol next hop, and in such cases, the set of BGP routes would reference the same indirect next hop.

Prior to Junos OS Release 17.2R1, the resolver module of the routing protocol process (rpd) resolved routes within the IBGP received routes in the following ways:

1. Partial route resolution—The protocol next hop is resolved based on helper routes, such as RSVP or IGP routes. The metric values are derived from the helper routes, and the next hop is referred to as the resolver forwarding next hop inherited from helper routes. These metric values are used for selecting routes in the routing information base (RIB), also known as the routing table.

2. Complete route resolution—The final next hop is derived and is referred to as the kernel routing table (KRT) forwarding next hop based on the forwarding export policy.

Starting in Junos OS Release 17.2R1, the resolver module of rpd is optimized to increase the throughput of inbound processing flow, accelerating the learning rate of RIB and FIB. With this enhancement, the route resolution is affected as follows:

- The partial and complete route resolution methods are triggered for each IBGP route, although each route might inherit the same resolved forwarding next hop or KRT forwarding next hops.

- The BGP path selection is deferred until complete route resolution is performed for network layer reachability information (NLRI) received from a BGP neighbor, which might not be the best route in the RIB after route selection.

The benefits of the rpd resolver optimization include:

- Lower RIB resolution lookup cost—The output of the resolved path is saved in a resolver cache, so that the same derived next hop and metric values can be inherited to another set of routes sharing the same path behavior instead of performing both partial and complete route resolution flow. This reduces the route resolution lookup cost by maintaining only the most frequent resolver state in a cache with limited depth.

- BGP route selection optimization—The BGP route selection algorithm is triggered twice for every IBGP route received—first, while adding the route in the RIB with the next hop as unusable, and

second, while adding the route with a resolved next hop in the RIB (after route resolution). This results in selecting the best route twice. With the resolver optimization, the route selection process is triggered in the receive flow only after getting the next-hop information from the resolver module.

- Internal caching to avoid frequent lookup—The resolver cache maintains the most frequent resolver state, and as a result, the lookup functionality, such as next-hop lookup and route lookup is done from the local cache.
- Path equivalence group—When different paths share the same forwarding state, or are received from the same protocol next hop, the paths can belong to one path equivalence group. This approach avoids the need to perform of complete route resolution for such paths. When a new path requires complete route resolution, it is first looked up in the path equivalence group database, which contains the resolved path output, such as indirect next hop and forwarding next hop.

SEE ALSO

BGP Routes Overview | **6**

Troubleshooting BGP Sessions | **1476**

Examine BGP Routes and Route Selection | **1492**

BGP Messages Overview

IN THIS SECTION

- Open Messages | **9**
- Update Messages | **9**
- Keepalive Messages | **10**
- Notification Messages | **10**
- Route-Refresh Messages | **10**

All BGP messages have the same fixed-size header, which contains a marker field that is used for both synchronization and authentication, a length field that indicates the length of the packet, and a type field that indicates the message type (for example, open, update, notification, keepalive, and so on).

This section discusses the following topics:

Open Messages

After a TCP connection is established between two BGP systems, they exchange BGP open messages to create a BGP connection between them. Once the connection is established, the two systems can exchange BGP messages and data traffic.

Open messages consist of the BGP header plus the following fields:

- Version—The current BGP version number is 4.
- Local AS number—You configure this by including the `autonomous-system` statement at the `[edit routing-options]` or `[edit logical-systems *logical-system-name* routing-options]` hierarchy level.
- Hold time—Proposed hold-time value. You configure the local hold time with the BGP `hold-time` statement.
- BGP identifier—IP address of the BGP system. This address is determined when the system starts and is the same for every local interface and every BGP peer. You can configure the BGP identifier by including the `router-id` statement at the `[edit routing-options]` or `[edit logical-systems *logical-system-name* routing-options]` hierarchy level. By default, BGP uses the IP address of the first interface it finds in the router.
- Parameter field length and the parameter itself—These are optional fields.

Update Messages

BGP systems send update messages to exchange network reachability information. BGP systems use this information to construct a graph that describes the relationships among all known ASs.

Update messages consist of the BGP header plus the following optional fields:

- Unfeasible routes length—Length of the withdrawn routes field
- Withdrawn routes—IP address prefixes for the routes being withdrawn from service because they are no longer deemed reachable
- Total path attribute length—Length of the path attributes field; it lists the path attributes for a feasible route to a destination
- Path attributes—Properties of the routes, including the path origin, the multiple exit discriminator (MED), the originating system's preference for the route, and information about aggregation, communities, confederations, and route reflection
- Network layer reachability information (NLRI)—IP address prefixes of feasible routes being advertised in the update message

Keepalive Messages

BGP systems exchange keepalive messages to determine whether a link or host has failed or is no longer available. Keepalive messages are exchanged often enough so that the hold timer does not expire. These messages consist only of the BGP header.

Notification Messages

BGP systems send notification messages when an error condition is detected. After the message is sent, the BGP session and the TCP connection between the BGP systems are closed. Notification messages consist of the BGP header plus the error code and subcode, and data that describes the error.

Route-Refresh Messages

BGP systems send route-refresh messages to a peer only if they have received the route refresh capability advertisement from the peer. A BGP system must advertise the route refresh capability to its peers using BGP capabilities advertisement if it wants to receive route-refresh messages. This optional message is sent to request dynamic, inbound, BGP route updates from BGP peers or to send outbound route updates to a BGP peer.

Route-refresh messages consist of the following fields:

- AFI—Address Family Identifier (16-bit).
- Res—Reserved (8-bit) field, which must be set to 0 by the sender and ignored by the receiver.
- SAFI—Subsequent Address Family Identifier (8-bit).

If a peer without the route-refresh capability receives a route-refresh request message from a remote peer, the receiver ignores the message.

SEE ALSO

Understanding BGP | 2

BGP Routes Overview | 6

Understanding BGP RIB sharding and BGP Update IO thread

BGP route processing usually has several pipeline stages such as receiving update, parsing update, creating route, resolving next-hop, applying a BGP peer group's export policy, forming per peer updates and sending updates to peers.

BGP RIB sharding splits a unified BGP RIB into several sub-RIBs and each sub-RIB handles a subset of BGP routes. Separate RPD thread termed BGP shard thread serves each sub-RIB by to achieve concurrency. BGP shard threads are responsible for all the BGP route processing pipeline stages with the exception of forming per peer updates and sending updates to peers. BGP shard threads receive the updates sent by peers from the BGP Update IO threads with the BGP Update IO threads hashing the prefixes in the updates and sends the updates to the applicable BGP shard threads based on the hash computation. BGP shard thread processes the configuration in the same manner as the RPD main thread, creates peers, groups, route-tables, and uses the configuration information for BGP route processing.

BGP Update IO threads are responsible for the tail end of this BGP pipeline, involving generating per peer updates for individual BGP group(s) and sending them to the peer(s). One update thread might serve one or more BGP groups. BGP Update IO threads construct updates for groups in parallel and independent of other groups that are being serviced by other update threads. This might offer significant convergence improvement in a write-heavy workload, that involves advertising to many peers spread across many groups. BGP Update IO threads are also responsible for writing to and reading from the BGP peers' TCP sockets which was previously provided by BGPIO threads (hence the suffix IO in BGP Update IO).

BGP Update IO threads can be configured independent of RIB sharding feature but are mandatory to use with RIB sharding, in order to achieve better prefix packing efficiency in outbound BGP update message. BGP sharding splits the RIB into several sub RIBs that are served by separate RPD threads. Hence, prefixes that could have gone into a single outbound update end up in different shards. To be able to construct BGP updates with prefixes with the same outgoing attribute that might belong to different RPD shard threads, all shard threads send compact advertisement information for prefixes to be advertised to an Update thread serving that BGP peer group. This allows the update thread, serving this BGP peer group, to pack prefixes with the same attributes, potentially belonging to different shards in the same outbound update message. This minimizes the number of updates to be advertised and thus helps improve convergence. Update IO thread manages local caches of peer, group, prefix, TSI and RIB containers.

BGP update thread and BGP RIB sharding are disabled by default. If you configure update-threading and rib-sharding on a routing engine, RPD creates update threads. By default, the number of update threads and shard threads created is the same as the number of CPU cores on the routing engine. Update threading is only supported on a 64 bit routing protocol process (rpd). Optionally, you can specify the number-of-threads you want to create by using `set update-threading <number-of-threads>` and `set rib-sharding <number-of-threads>` statements at the `[edit system processes routing bgp]` hierarchy level. For BGP update

thread, the range is currently 1 through 128 and for BGP RIB sharding, the range is currently 1 through 31.

When you configure NSR for the BGP RIB sharding and BGP Update IO features, the backup RPD creates the same number of BGP shard and BGP Update IO threads in the backup routing engine. The backup RPD BGP Update IO threads read the replicated BGP update, other messages received from the peers as well as replicated BGP update, and other messages sent to the peers. Based on hashing of prefixes, the backup RPD BGP Update IO threads send these BGP messages to the applicable BGP shard and RPD main threads. The BGP shard and the RPD main threads in the backup RPD creates the received and advertised route state using these replicated BGP messages. When the primary routing engine fails, the backup routing-engine becomes the primary routing engine and the backup RPD becomes the primary RPD seamlessly without impacting the BGP sessions with the peers.

Understanding BGP Path Selection

IN THIS SECTION

- Routing Table Path Selection | **14**
- BGP Table path selection | **16**
- Effects of Advertising Multiple Paths to a Destination | **16**

For each prefix in the routing table, the routing protocol process selects a single best path. After the best path is selected, the route is installed in the routing table. The best path becomes the active route if the same prefix is not learned by a protocol with a lower (more preferred) global preference value, also known as the administrative distance. The algorithm for determining the active route is as follows:

1. Verify that the next hop can be resolved.
2. Choose the path with the lowest preference value (routing protocol process preference).

   Routes that are not eligible to be used for forwarding (for example, because they were rejected by routing policy or because a next hop is inaccessible) have a preference of –1 and are never chosen.
3. Prefer the path with higher local preference.

   For non-BGP paths, choose the path with the lowest **preference2** value.
4. If the accumulated interior gateway protocol (AIGP) attribute is enabled, add the IGP metric and prefer the path with the lower AIGP attribute.

5. Prefer the path with the shortest autonomous system (AS) path value (skipped if the `as-path-ignore` statement is configured).

   A confederation segment (sequence or set) has a path length of 0. An AS set has a path length of 1.

6. Prefer the route with the lower origin code.

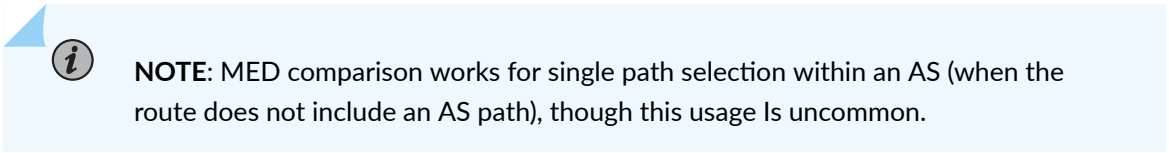
   Routes learned from an IGP have a lower origin code than those learned from an exterior gateway protocol (EGP), and both have lower origin codes than incomplete routes (routes whose origin is unknown).

7. Prefer the path with the lowest multiple exit discriminator (MED) metric.

   Depending on whether nondeterministic routing table path selection behavior is configured, there are two possible cases:

   - If nondeterministic routing table path selection behavior is not configured (that is, if the `path-selection cisco-nondeterministic` statement is not included in the BGP configuration), for paths with the same neighboring AS numbers at the front of the AS path, prefer the path with the lowest MED metric. To always compare MEDs whether or not the peer ASs of the compared routes are the same, include the `path-selection always-compare-med` statement.
   - If nondeterministic routing table path selection behavior is configured (that is, the `path-selection cisco-nondeterministic` statement is included in the BGP configuration), prefer the path with the lowest MED metric.

   Confederations are not considered when determining neighboring ASs. A missing MED metric is treated as if a MED were present but zero.

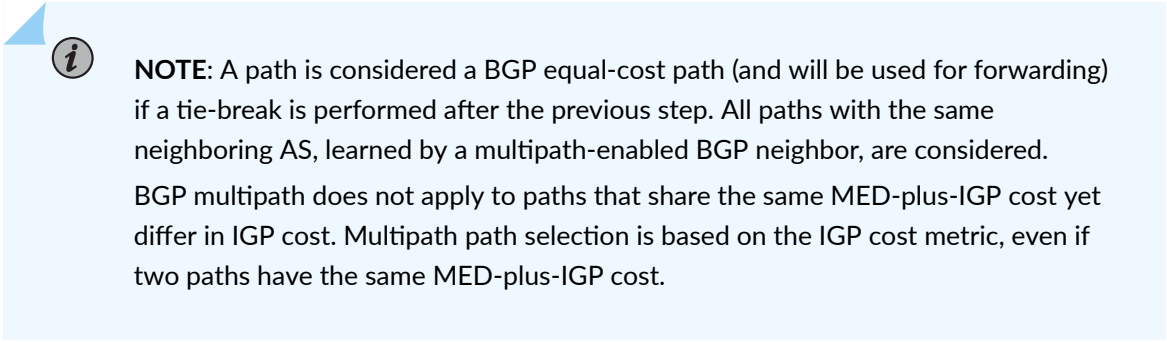
   NOTE: MED comparison works for single path selection within an AS (when the route does not include an AS path), though this usage Is uncommon.

   By default, only the MEDs of routes that have the same peer autonomous systems (ASs) are compared. You can configure routing table path selection options to obtain different behaviors.

8. Prefer strictly internal paths, which include IGP routes and locally generated routes (static, direct, local, and so forth).

9. Prefer strictly external BGP (EBGP) paths over external paths learned through internal BGP (IBGP) sessions.

10. Prefer the path whose next hop is resolved through the IGP route with the lowest metric. BGP routes that are resolved through IGP are preferred over unreachable or rejected routes.



NOTE: A path is considered a BGP equal-cost path (and will be used for forwarding) if a tie-break is performed after the previous step. All paths with the same neighboring AS, learned by a multipath-enabled BGP neighbor, are considered.

BGP multipath does not apply to paths that share the same MED-plus-IGP cost yet differ in IGP cost. Multipath path selection is based on the IGP cost metric, even if two paths have the same MED-plus-IGP cost.

11. If both paths are external, prefer the oldest path, in other words, the path that was learned first. This is done to minimize route-flapping. This rule is not used if any one of the following conditions is true:

    - **path-selection external-router-id** is configured.
    - Both peers have the same router ID.
    - Either peer is a confederation peer.
    - Neither path is the current active path.

12. Prefer a primary route over a secondary route. A primary route is one that belongs to the routing table. A secondary route is one that is added to the routing table through an export policy.

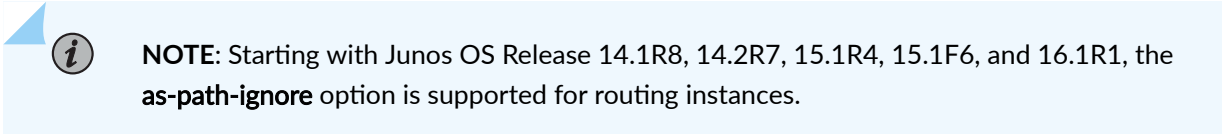
13. Prefer the path from the peer with the lowest router ID. For any path with an originator ID attribute, substitute the originator ID for the router ID during router ID comparison.

14. Prefer the path with the shortest cluster list length. The length is 0 for no list.

15. Prefer the path from the peer with the lowest peer IP address.

Routing Table Path Selection

The shortest AS path step of the algorithm, by default, evaluates the length of the AS path and determines the active path. You can configure an option that enables Junos OS to skip this step of the algorithm by including the **as-path-ignore** option.



NOTE: Starting with Junos OS Release 14.1R8, 14.2R7, 15.1R4, 15.1F6, and 16.1R1, the **as-path-ignore** option is supported for routing instances.

The routing process path selection takes place before BGP hands off the path to the routing table to makes its decision. To configure routing table path selection behavior, include the `path-selection` statement:

```
path-selection {
    (always-compare-med | cisco-non-deterministic | external-router-id);
    as-path-ignore;
    l2vpn-use-bgp-rules;
    med-plus-igp {
        igp-multiplier number;
        med-multiplier number;
    }
}
```

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

Routing table path selection can be configured in one of the following ways:

- Emulate the Cisco IOS default behavior (**cisco-non-deterministic**). This mode evaluates routes in the order that they are received and does not group them according to their neighboring AS. With `cisco-non-deterministic` mode, the active path is always first. All inactive, but eligible, paths follow the active path and are maintained in the order in which they were received, with the most recent path first. Ineligible paths remain at the end of the list.

  As an example, suppose you have three path advertisements for the 192.168.1.0 /24 route:

  - Path 1—learned through EBGP; AS Path of 65010; MED of 200
  - Path 2—learned through IBGP; AS Path of 65020; MED of 150; IGP cost of 5
  - Path 3—learned through IBGP; AS Path of 65010; MED of 100; IGP cost of 10

  These advertisements are received in quick succession, within a second, in the order listed. Path 3 is received most recently, so the routing device compares it against path 2, the next most recent advertisement. The cost to the IBGP peer is better for path 2, so the routing device eliminates path 3 from contention. When comparing paths 1 and 2, the routing device prefers path 1 because it is received from an EBGP peer. This allows the routing device to install path 1 as the active path for the route.

  NOTE: We do not recommend using this configuration option in your network. It is provided solely for interoperability to allow all routing devices in the network to make consistent route selections.

- Always comparing MEDs whether or not the peer ASs of the compared routes are the same (**always-compare-med**).
- Override the rule that If both paths are external, the currently active path is preferred (**external-router-id**). Continue with the next step (Step "12" on page 14) in the path-selection process.
- Adding the IGP cost to the next-hop destination to the MED value before comparing MED values for path selection (`med-plus-igp`).

  BGP multipath does not apply to paths that share the same MED-plus-IGP cost, yet differ in IGP cost. Multipath path selection is based on the IGP cost metric, even if two paths have the same MED-plus-IGP cost.

BGP Table path selection

The following parameters are followed for BGP's path selection:

1. Prefer the highest local-preference value.
2. Prefer the shortest AS-path length.
3. Prefer the lowest origin value.
4. Prefer the lowest MED value.
5. Prefer routes learned from an EBGP peer over an IBGP peer.
6. Prefer best exit from AS.
7. For EBGP-received routes, prefer the current active route.
8. Prefer routes from the peer with the lowest Router ID.
9. Prefer paths with the shortest cluster length.
10. Prefer routes from the peer with the lowest peer IP address. Steps 2, 6 and 12 are the RPD criteria.

Effects of Advertising Multiple Paths to a Destination

BGP advertises only the active path, unless you configure BGP to advertise multiple paths to a destination.

Suppose a routing device has in its routing table four paths to a destination and is configured to advertise up to three paths (**add-path send path-count 3**). The three paths are chosen based on path selection criteria. That is, the three best paths are chosen in path-selection order. The best path is the active path. This path is removed from consideration and a new best path is chosen. This process is repeated until the specified number of paths is reached.

SEE ALSO

- Example: Ignoring the AS Path Attribute When Selecting the Best Path
- Examples: Configuring BGP MED
- Example: Advertising Multiple BGP Paths to a Destination

Supported Standards for BGP

Junos OS substantially supports the following RFCs and Internet drafts, which define standards for IP version 4 (IPv4) BGP.

For a list of supported IP version 6 (IPv6) BGP standards, see *Supported IPv6 Standards*.

Junos OS BGP supports authentication for protocol exchanges (MD5 authentication).

- RFC 1745, *BGP4/IDRP for IP—OSPF Interaction*
- RFC 1772, *Application of the Border Gateway Protocol in the Internet*
- RFC 1997, *BGP Communities Attribute*
- RFC 2283, *Multiprotocol Extensions for BGP-4*
- RFC 2385, *Protection of BGP Sessions via the TCP MD5 Signature Option*
- RFC 2439, *BGP Route Flap Damping*
- RFC 2545, *Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*
- RFC 2796, *BGP Route Reflection – An Alternative to Full Mesh IBGP*
- RFC 2858, *Multiprotocol Extensions for BGP-4*
- RFC 2918, *Route Refresh Capability for BGP-4*
- RFC 3065, *Autonomous System Confederations for BGP*
- RFC 3107, *Carrying Label Information in BGP-4*
- RFC 3345, *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*
- RFC 3392, *Capabilities Advertisement with BGP-4*
- RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*
- RFC 4273, *Definitions of Managed Objects for BGP-4*

- RFC 4360, *BGP Extended Communities Attribute*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4456, *BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*
- RFC 4486, *Subcodes for BGP Cease Notification Message*
- RFC 4576, *Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*
- RFC 4632, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*
- RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*
- RFC 4724, *Graceful Restart Mechanism for BGP*
- RFC 4760, *Multiprotocol Extensions for BGP-4*
- RFC 4781, *Graceful Restart Mechanism for BGP with MPLS*
- RFC 4798, *Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*

  Option 4b (eBGP redistribution of labeled IPv6 routes from AS to neighboring AS) is not supported.
- RFC 4893, *BGP Support for Four-octet AS Number Space*
- RFC 5004, *Avoid BGP Best Path Transitions from One External to Another*
- RFC 5065, *Autonomous System Confederations for BGP*
- RFC 5082, *The Generalized TTL Security Mechanism (GTSM)*
- RFC 5291, *Outbound Route Filtering Capability for BGP-4 (partial support)*
- RFC 5292, *Address-Prefix-Based Outbound Route Filter for BGP-4 (partial support)*

  Devices running Junos OS can receive prefix-based ORF messages.
- RFC 5396, *Textual Representation of Autonomous System (AS) Numbers*
- RFC 5492, *Capabilities Advertisement with BGP-4*
- RFC 5512, *The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute*
- RFC 5549, *Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop*

- RFC 8955, *Dissemination of Flow Specification Rules*
- RFC 8956, *Dissemination of Flow Specification Rules for IPv6*
- RFC 5668, *4-Octet AS Specific BGP Extended Community*
- RFC 5701, *IPv6 Address Specific BGP Extended Community Attribute*
- RFC 5925, *The TCP Authentication Option*
- RFC 6286, *Autonomous-System-Wide Unique BGP Identifier for BGP-4- fully compliant*
- RFC 6368, *Internal BGP as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 6774, *Distribution of Diverse BGP Paths*
- RFC 6793, *BGP Support for Four-Octet Autonomous System (AS) Number Space*
- RFC 6810, *The Resource Public Key Infrastructure (RPKI) to Router Protocol*
- RFC 6811, *BGP Prefix Origin Validation*
- RFC 6996, *Autonomous System (AS) Reservation for Private Use*
- RFC 7300, *Reservation of Last Autonomous System (AS) Numbers*
- RFC 7311, *The Accumulated IGP Metric Attribute for BGP*
- RFC 7404, *Using Only Link-Local Addressing inside an IPv6 Network*
- RFC 7432, *BGP MPLS-Based Ethernet VPN (eVPN)*
- RFC 7606, *Revised Error Handling for BGP UPDATE Messages*
- RFC 7611, *BGP ACCEPT_OWN Community Attribute*

  We support the RFC by enabling Juniper routers to accept routes received from a route reflector with the `accept-own` community value.

- RFC 7752, *North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*
- RFC 7854, *BGP Monitoring Protocol (BMP)*
- RFC 7911, *Advertisement of Multiple Paths in BGP*
- RFC 8097, *BGP Prefix Origin Validation State Extended Community*
- RFC 8210, *The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1*

- RFC 8212, *Default External BGP (EBGP) Route Propagation Behavior without Policies- fully compliant*

  Exceptions:

  The behaviors in RFC 8212 are not implemented by default in order to avoid disruption of existing customer configuration. The default behavior is still kept to accept and advertise all routes with regard to EBGP peers.

- RFC 8277, *Using BGP to Bind MPLS Labels to Address Prefixes*
- RFC 8326, *Graceful BGP session Shutdown*
- RFC 8481, *Clarifications to BGP Origin Validation Based on Resource Public Key Infrastructure (RPKI)*
- RFC 8538, *Notification Message Support for BGP Graceful Restart*
- RFC 8571, *BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions*
- RFC 8584, *Framework for Ethernet VPN Designated Forwarder Election Extensibility*
- RFC 8642, *Policy Behavior for Well-Known BGP Communities*
- RFC 8669, *Segment Routing Prefix Segment Identifier Extensions for BGP*
- RFC 8810, *Revision to Capability Codes Registration Procedures*
- RFC 8814 *Signaling Maximum SID Depth (MSD) Using the Border Gateway Protocol - Link State (partial support)*
- RFC 8950, *Advertising IPv4 Network Layer Reachability Information (NLRI) with an IPv6 Next Hop*
- RFC 8955
- RFC 8956
- RFC 9003, *Extended BGP Administrative Shutdown Communication*
- RFC 9012, *The BGP Tunnel Encapsulation Attribute*
- RFC 9029, *Updates to the Allocation Policy for the Border Gateway Protocol - Link State (BGP-LS) Parameters Registries*
- RFC 9069, *Support for Local RIB in the BGP Monitoring Protocol (BMP)*
- RFC 9085, *Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing*

- RFC 9117, *Revised Validation procedure for Dissemination of BGP Flow Specifications. This enables the Flow Specifications to be originated within the same autonomous system as the BGP peer performing the validation.*
- RFC 9384, *A BGP Cease NOTIFICATION Subcode for Bidirectional Forwarding Detection (BFD)*
- Internet draft draft-idr-rfc8203bis-00, *BGP Administrative Shutdown Communication* (expires October 2018)
- Internet draft draft-ietf-grow-bmp-adj-rib-out-01, *Support for Adj-RIB-Out in BGP Monitoring Protocol (BMP)* (expires September 3, 2018)
- Internet draft draft-ietf-idr-aigp-06, *The Accumulated IGP Metric Attribute for BGP* (expires December 2011)
- Internet draft draft-ietf-idr-as0-06, *Codification of AS 0 processing* (expires February 2013)
- Internet draft draft-ietf-idr-link-bandwidth-06.txt, *BGP Link Bandwidth Extended Community* (expires July 2013)
- Internet draft draft-ietf-sidr-origin-validation-signaling-00, *BGP Prefix Origin Validation State Extended Community (partial support)* (expires May 2011)

  The extended community (origin validation state) is supported in Junos OS routing policy. The specified change in the route selection procedure is not supported.
- Internet draft draft-kato-bgp-ipv6-link-local-00.txt, *BGP4+ Peering Using IPv6 Link-local Address*

The following RFCs and Internet draft do not define standards, but provide information about BGP and related technologies. The IETF classifies them variously as “Experimental” or “Informational.”

- RFC 1965, *Autonomous System Confederations for BGP*
- RFC 1966, *BGP Route Reflection—An alternative to full mesh IBGP*
- RFC 2270, *Using a Dedicated AS for Sites Homed to a Single Provider*
- RFC 3345, *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*
- RFC 3562, *Key Management Considerations for the TCP MD5 Signature Option*
- Internet draft draft-ietf-ngtrans-bgp-tunnel-04.txt, *Connecting IPv6 Islands across IPv4 Clouds with BGP* (expires July 2002)

SEE ALSO

> *Supported IPv6 Standards*

Accessing Standards Documents on the Internet

Change History Table

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

Release	Description
17.2R1	Starting in Junos OS Release 17.2R1, the resolver module of rpd is optimized to increase the throughput of inbound processing flow, accelerating the learning rate of RIB and FIB.
14.1R8	Starting with Junos OS Release 14.1R8, 14.2R7, 15.1R4, 15.1F6, and 16.1R1, the **as-path-ignore** option is supported for routing instances.

2

CHAPTER

Basic BGP Configurations

IN THIS CHAPTER

- BGP Configuration Overview | **24**
- BGP Peering Sessions | **24**
- BGP Route Prioritization | **101**
- BGP Auto-Discovered Neighbors | **122**

BGP Configuration Overview

To configure the device as a node in a BGP network:

1. Configure network interfaces. See the Ethernet Interfaces User Guide for Routing Devices.
2. Configure point-to-point peering sessions. See "Example: Configuring External BGP Point-to-Point Peer Sessions" on page 27.
3. Configure IBGP sessions between peers. See "Example: Configuring Internal BGP Peer Sessions" on page 63.
4. Configure BGP session attributes such as the autonomous systems for the BGP peers. See "Autonomous Systems for BGP Sessions" on page 141
5. Configure a routing policy to advertise the BGP routes.
6. (Optional) Configure route reflector clusters. See "Example: Configuring a Route Reflector" on page 1182.
7. (Optional) Subdivide autonomous systems (ASs). See "Example: Configuring BGP Confederations" on page 1223.
8. (Optional) Assign a router ID to each routing device running BGP.
9. (Optional) Configure a local preference to direct all outbound AS traffic to a specific peer. See "Example: Configuring the Local Preference Value for BGP Routes" on page 297.
10. (Optional) Configure routing table path selection options that define different ways to compare multiple exit discriminators (MEDs). See "Understanding BGP Path Selection" on page 12.

RELATED DOCUMENTATION

Understanding BGP | 2

BGP Peering Sessions

IN THIS SECTION

- Advantages of Using External BGP Peer Groups | 25
- Understanding External BGP Peering Sessions | 26
- Example: Configuring External BGP Point-to-Point Peer Sessions | 27

- Example: Configuring External BGP on Logical Systems with IPv6 Interfaces | **39**
- Understanding Internal BGP Peering Sessions | **61**
- Example: Configuring Internal BGP Peer Sessions | **63**
- Example: Configuring Internal BGP Peering Sessions on Logical Systems | **80**
- Overview: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same Link-Local Address (IPv6) | **94**
- Example: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same IPv6 Link-Local Address | **95**

Advantages of Using External BGP Peer Groups

BGP is the only routing protocol in use today that is suited to carry all of the routes in the Internet. This is largely because BGP runs on top of TCP and can make use of TCP flow control. In contrast, the internal gateway protocols (IGPs) do not have flow control. When IGPs have too much route information, they begin to churn. When BGP has a neighboring speaker that is sending information too quickly, BGP can throttle down the neighbor by delaying TCP acknowledgments.

Another benefit of BGP is that (like IS-IS) it uses type, length, value (TLV) tuples and network layer reachability information (NLRI) that provide seemingly endless extensibility without the need for the underlying protocol to be altered.

In Junos OS, BGP is completely policy driven. The operator must explicitly configure neighbors to peer with and explicity accept routes into BGP. Further, routing policy is used to filter and modify routing information. Thus, routing policies provide complete administrative control over the routing tables.

The preferred way to configure a large number of BGP peer neighbors is to configure peer groups consisting of multiple neighbors per group.

As the number of external BGP (EBGP) groups increases, the ability to support a large number of BGP sessions might become a CPU and memory resource scaling issue. Supporting fewer EBGP groups generally scales better than supporting a large number of EBGP groups. This becomes more evident in the case of hundreds of EBGP groups when compared with a few EBGP groups with multiple peers in each group. The reason for this scaling behavior is that Junos OS has data structures that occur on a per route-per group basis. When you add a group, you multiply those numbers and decrease the amount of memory available.

BGP peering creates mutually beneficial traffic exchange relationships between two independent autonomous systems (ASs). It is especially useful at service provider exchange points. This relationship has the primary benefit of reducing transit costs and equipment resources for both networks. Other

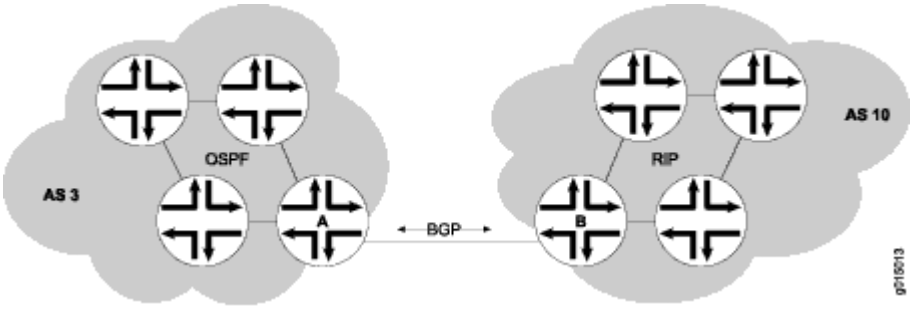
potential benefits of creating BGP peer groups include reducing the complexity of the BGP configuration and increasing route redundancy by reducing the dependence on transit providers.

BGP peering can be used to create point-to-point traffic exchanges between two remote networks, such as a remote office and the company headquarters. It can also be used to quickly connect two disparate networks, such as between two merged offices.

Understanding External BGP Peering Sessions

To establish point-to-point connections between peer autonomous systems (ASs), you configure a BGP session on each interface of a point-to-point link. Generally, such sessions are made at network exit points with neighboring hosts outside the AS. Figure 2 on page 26 shows an example of a BGP peering session.

Figure 2: BGP Peering Session



In Figure 2 on page 26, Router A is a gateway router for AS 3, and Router B is a gateway router for AS 10. For traffic internal to either AS, an interior gateway protocol (IGP) is used (OSPF, for instance). To route traffic between peer ASs, a BGP session is used.

You arrange BGP routing devices into groups of peers. Different peer groups can have different group types, AS numbers, and route reflector cluster identifiers.

To define a BGP group that recognizes only the specified BGP systems as peers, statically configure all the system's peers by including one or more `neighbor` statements. The peer neighbor's address can be either an IPv6 or IPv4 address.

After the BGP peers are established, non-BGP routes are not automatically advertised by the BGP peers. At each BGP-enabled device, policy configuration is required to export the local, static, or IGP-learned routes into the BGP RIB and then advertise them as BGP routes to the other peers. BGP's advertisement policy, by default, does not advertise any non-BGP routes (such as local routes) to peers.

NOTE: On SRX Series Firewalls, you must enable the expected host-inbound traffic on the specified interfaces or all interfaces of the zone. Otherwise inbound traffic destined to this device is dropped by default.

For example, to allow BGP traffic on a specific zone of your SRX Series Firewall, use the following step:

```
[edit]
user@host# set security zones security-zone trust host-inbound-traffic protocols bgp
```

(All interfaces)

```
[edit]
user@host# set security zones security-zone trust interfaces ge-0/0/1.0 host-inbound-
traffic protocols bgp
```

(Specified interface)

SEE ALSO

Understanding BGP | 2

Example: Configuring Internal BGP Peer Sessions | 63

forwarding-options (Security)

Example: Configuring External BGP Point-to-Point Peer Sessions

IN THIS SECTION

- Requirements | 28
- Overview | 28
- Configuration | 29
- Verification | 33

This example shows how to configure BGP point-to-point peer sessions.

Requirements

Before you begin, if the default BGP policy is not adequate for your network, configure routing policies to filter incoming BGP routes and to advertise BGP routes.

Overview

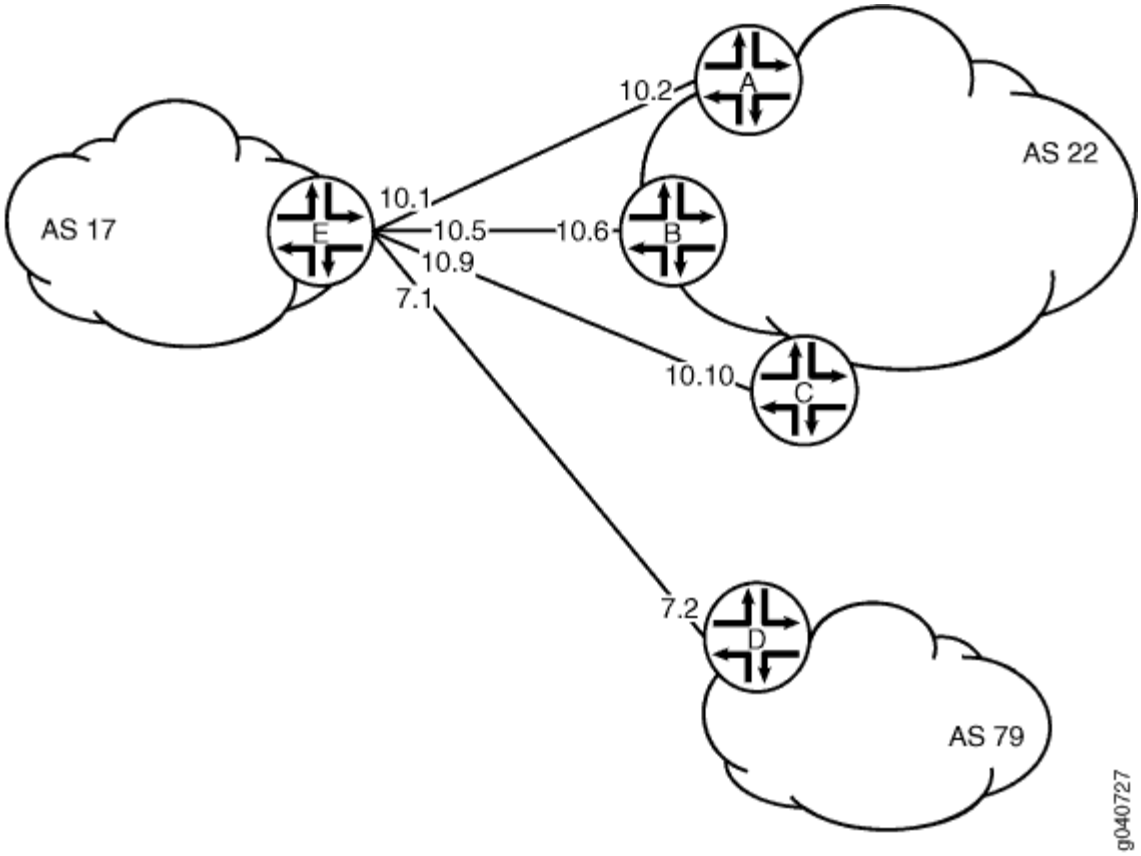
IN THIS SECTION

- Topology | **29**

Figure 3 on page 29 shows a network with BGP peer sessions. In the sample network, Device E in AS 17 has BGP peer sessions to a group of peers called `external-peers`. Peers A, B, and C reside in AS 22 and have IP addresses 10.10.10.2, 10.10.10.6, and 10.10.10.10. Peer D resides in AS 79, at IP address 10.21.7.2. This example shows the configuration on Device E.

Topology

Figure 3: Typical Network with BGP Peer Sessions



Configuration

IN THIS SECTION

- Procedure | 30

Procedure

CLI Quick Configuration

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

```
set interfaces ge-1/2/0 unit 0 description to-A
set interfaces ge-1/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces ge-0/0/1 unit 5 description to-B
set interfaces ge-0/0/1 unit 5 family inet address 10.10.10.5/30
set interfaces ge-0/1/0 unit 9 description to-C
set interfaces ge-0/1/0 unit 9 family inet address 10.10.10.9/30
set interfaces ge-1/2/1 unit 21 description to-D
set interfaces ge-1/2/1 unit 21 family inet address 10.21.7.1/30
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 22
set protocols bgp group external-peers neighbor 10.10.10.2
set protocols bgp group external-peers neighbor 10.10.10.6
set protocols bgp group external-peers neighbor 10.10.10.10
set protocols bgp group external-peers neighbor 10.21.7.2 peer-as 79
set routing-options autonomous-system 17
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the interfaces to Peers A, B, C, and D.

```
[edit interfaces]
user@E# set ge-1/2/0 unit 0 description to-A
user@E# set ge-1/2/0 unit 0 family inet address 10.10.10.1/30
user@E# set ge-0/0/1 unit 5 description to-B
user@E# set ge-0/0/1 unit 5 family inet address 10.10.10.5/30
user@E# set ge-0/1/0 unit 9 description to-C
user@E# set ge-0/1/0 unit 9 family inet address 10.10.10.9/30
```

```
user@E# set ge-1/2/1 unit 21 description to-D
user@E# set ge-1/2/1 unit 21 family inet address 10.21.7.1/30
```

2. Set the autonomous system (AS) number.

```
[edit routing-options]
user@E# set autonomous-system 17
```

3. Create the BGP group, and add the external neighbor addresses.

```
[edit protocols bgp group external-peers]
user@E# set neighbor 10.10.10.2
user@E# set neighbor 10.10.10.6
user@E# set neighbor 10.10.10.10
```

4. Specify the autonomous system (AS) number of the external AS.

```
[edit protocols bgp group external-peers]
user@E# set peer-as 22
```

5. Add Peer D, and set the AS number at the individual neighbor level.

   The neighbor configuration overrides the group configuration. So, while `peer-as 22` is set for all the other neighbors in the group, `peer-as 79` is set for neighbor 10.21.7.2.

```
[edit protocols bgp group external-peers]
user@E# set neighbor 10.21.7.2 peer-as 79
```

6. Set the peer type to external BGP (EBGP).

```
[edit protocols bgp group external-peers]
user@E# set type external
```

Results

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

```
[edit]
user@E# show interfaces
ge-1/2/0 {
    unit 0 {
        description to-A;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
ge-0/0/1 {
    unit 5 {
        description to-B;
        family inet {
            address 10.10.10.5/30;
        }
    }
}
ge-0/1/0 {
    unit 9 {
        description to-C;
        family inet {
            address 10.10.10.9/30;
        }
    }
}
ge-1/2/1 {
    unit 21 {
        description to-D;
        family inet {
            address 10.21.7.1/30;
        }
```

```
    }
}
```

```
[edit]
user@E# show protocols
bgp {
    group external-peers {
        type external;
        peer-as 22;
        neighbor 10.10.10.2;
        neighbor 10.10.10.6;
        neighbor 10.10.10.10;
        neighbor 10.21.7.2 {
            peer-as 79;
        }
    }
}
```

```
[edit]
user@E# show routing-options
autonomous-system 17;
```

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

Verification

IN THIS SECTION

- Verifying BGP Neighbors | 34
- Verifying BGP Groups | 37
- Verifying BGP Summary Information | 38

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is active for each neighbor address.

Action

From operational mode, run the `show bgp neighbor` command.

```
user@E> show bgp neighbor
Peer: 10.10.10.2+179 AS 22     Local: 10.10.10.1+65406 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.10.10.2       Local ID: 10.10.10.1       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: ge-1/2/0.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
```

```
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 10   Sent 6    Checked 1
  Input messages:  Total 8522   Updates 1       Refreshes 0     Octets 161922
  Output messages: Total 8433   Updates 0       Refreshes 0     Octets 160290
  Output Queue[0]: 0

Peer: 10.10.10.6+54781 AS 22   Local: 10.10.10.5+179 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.10.10.6       Local ID: 10.10.10.1       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  Local Interface: ge-0/0/1.5
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 12   Sent 6    Checked 33
  Input messages:  Total 8527   Updates 1       Refreshes 0     Octets 162057
  Output messages: Total 8430   Updates 0       Refreshes 0     Octets 160233
  Output Queue[0]: 0
```

```
Peer: 10.10.10.10+55012 AS 22  Local: 10.10.10.9+179 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.10.10.10      Local ID: 10.10.10.1       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 2
  BFD: disabled, down
  Local Interface: fe-0/1/0.9
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 15   Sent 6    Checked 37
  Input messages:  Total 8527   Updates 1       Refreshes 0     Octets 162057
  Output messages: Total 8429   Updates 0       Refreshes 0     Octets 160214
  Output Queue[0]: 0

Peer: 10.21.7.2+61867 AS 79    Local: 10.21.7.1+179 AS 17
  Type: External    State: Established    Flags: <ImportEval Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
```

```
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.21.7.2        Local ID: 10.10.10.1       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 3
  BFD: disabled, down
  Local Interface: ge-1/2/1.21
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 79)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 28   Sent 24   Checked 47
  Input messages:  Total 8521   Updates 1       Refreshes 0     Octets 161943
  Output messages: Total 8427   Updates 0       Refreshes 0     Octets 160176
  Output Queue[0]: 0
```

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From operational mode, run the `show bgp group` command.

```
user@E> show bgp group
Group Type: External                              Local AS: 17
  Name: external-peers  Index: 0                  Flags: <>
  Holdtime: 0
  Total peers: 4        Established: 4
  10.10.10.2+179
  10.10.10.6+54781
  10.10.10.10+55012
  10.21.7.2+61867
  inet.0: 0/0/0/0

Groups: 1  Peers: 4    External: 4    Internal: 0    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 0          0          0          0          0          0
```

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From operational mode, run the `show bgp summary` command.

```
user@E> show bgp summary
Groups: 1 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.10.10.2               22       8559       8470       0       0 2d 16:12:56
0/0/0/0              0/0/0/0
10.10.10.6               22       8566       8468       0       0 2d 16:12:12
0/0/0/0              0/0/0/0
10.10.10.10              22       8565       8466       0       0 2d 16:11:31
```

```
0/0/0/0              0/0/0/0
10.21.7.2               79       8560       8465       0       0 2d 16:10:58
0/0/0/0              0/0/0/0
```

SEE ALSO

- Routing Policies, Firewall Filters, and Traffic Policers User Guide
- Understanding External BGP Peering Sessions | 26
- Example: Configuring Internal BGP Peer Sessions | 63
- BGP Configuration Overview | 24

Example: Configuring External BGP on Logical Systems with IPv6 Interfaces

IN THIS SECTION

- Requirements | 39
- Overview | 40
- Configuration | 42
- Verification | 54

This example shows how to configure external BGP (EBGP) point-to-point peer sessions on logical systems with IPv6 interfaces.

Requirements

In this example, no special configuration beyond device initialization is required.

Overview

IN THIS SECTION

- Topology | 41

Junos OS supports EBGP peer sessions by means of IPv6 addresses. An IPv6 peer session can be configured when an IPv6 address is specified in the `neighbor` statement. This example uses EUI-64 to generate IPv6 addresses that are automatically applied to the interfaces. An EUI-64 address is an IPv6 address that uses the IEEE EUI-64 format for the interface identifier portion of the address (the last 64 bits).

NOTE: Alternatively, you can configure EBGP sessions using manually assigned 128-bit IPv6 addresses.

If you use 128-bit link-local addresses for the interfaces, you must include the `local-interface` statement. This statement is valid only for 128-bit IPv6 link-local addresses and is mandatory for configuring an IPv6 EBGP link-local peer session.

Configuring EBGP peering using link-local addresses is only applicable for directly connected interfaces. There is no support for multihop peering.

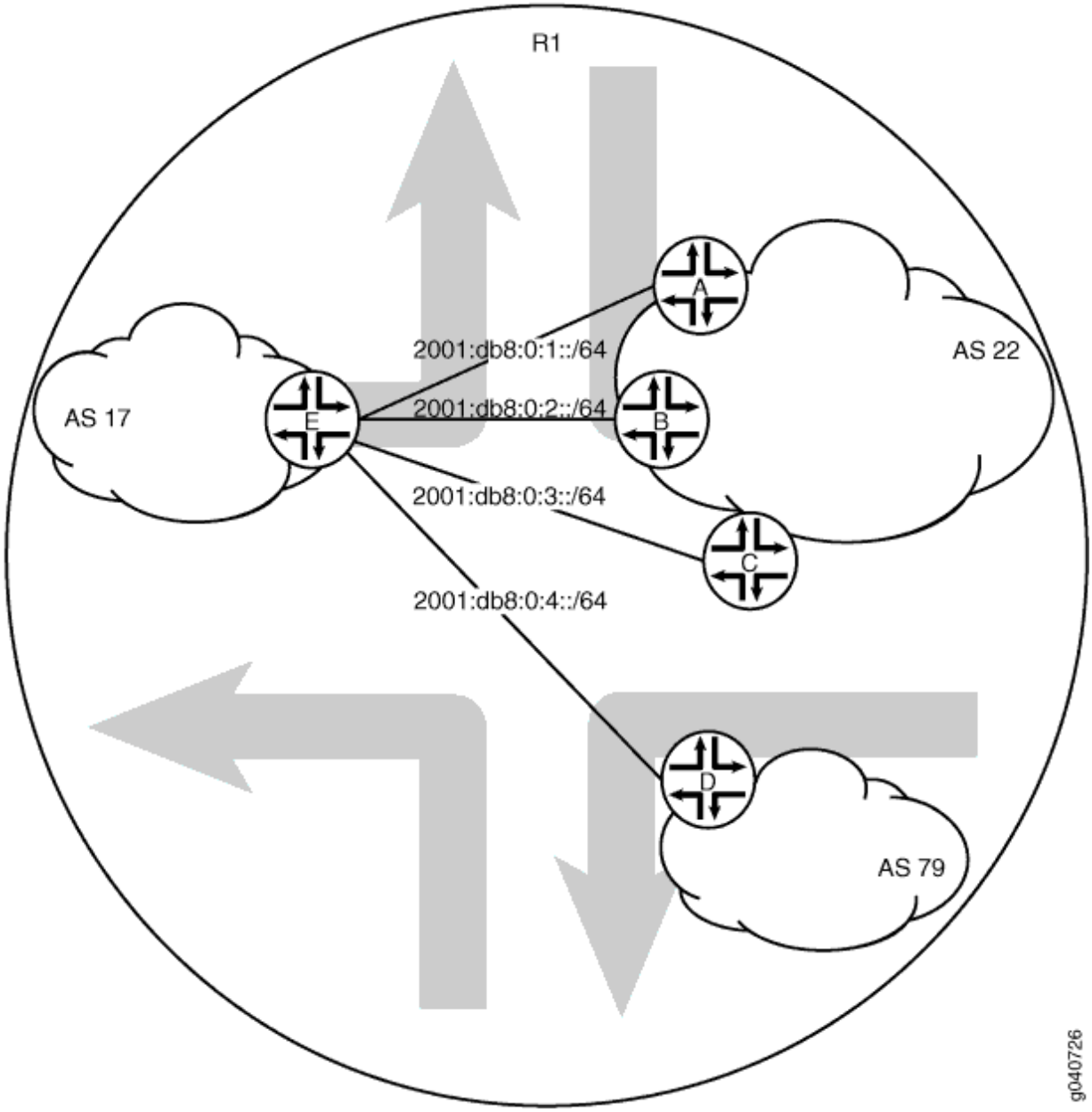
After your interfaces are up, you can use the `show interfaces terse` command to view the EUI-64-generated IPv6 addresses on the interfaces. You must use these generated addresses in the BGP `neighbor` statements. This example demonstrates the full end-to-end procedure.

In this example, Frame Relay interface encapsulation is applied to the logical tunnel (**lt**) interfaces. This is a requirement because only Frame Relay encapsulation is supported when IPv6 addresses are configured on the **lt** interfaces.

Figure 4 on page 41 shows a network with BGP peer sessions. In the sample network, Router R1 has five logical systems configured. Device E in autonomous system (AS) 17 has BGP peer sessions to a group of peers called **external-peers**. Peers A, B, and C reside in AS 22. This example shows the step-by-step configuration on Logical System A and Logical System E.

Topology

Figure 4: Typical Network with BGP Peer Sessions



Configuration

IN THIS SECTION

- Procedure | 42
- Configuring the External BGP Sessions | 47

Procedure

CLI Quick Configuration

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

Device A

```
set logical-systems A interfaces lt-0/1/0 unit 1 description to-E
set logical-systems A interfaces lt-0/1/0 unit 1 encapsulation frame-relay
set logical-systems A interfaces lt-0/1/0 unit 1 dlci 1
set logical-systems A interfaces lt-0/1/0 unit 1 peer-unit 25
set logical-systems A interfaces lt-0/1/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set logical-systems A interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
set logical-systems A protocols bgp group external-peers type external
set logical-systems A protocols bgp group external-peers peer-as 17
set logical-systems A protocols bgp group external-peers neighbor 2001:db8:0:1:2a0:a502:0:19da
set logical-systems A protocols bgp group external-peers family inet6 unicast
set logical-systems A routing-options router-id 172.16.1.1
set logical-systems A routing-options autonomous-system 22
```

Device B

```
set logical-systems B interfaces lt-0/1/0 unit 6 description to-E
set logical-systems B interfaces lt-0/1/0 unit 6 encapsulation frame-relay
set logical-systems B interfaces lt-0/1/0 unit 6 dlci 6
set logical-systems B interfaces lt-0/1/0 unit 6 peer-unit 5
set logical-systems B interfaces lt-0/1/0 unit 6 family inet6 address 2001:db8:0:2::/64 eui-64
set logical-systems B interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
```

```
set logical-systems B protocols bgp group external-peers type external
set logical-systems B protocols bgp group external-peers peer-as 17
set logical-systems B protocols bgp group external-peers neighbor 2001:db8:0:2:2a0:a502:0:5da
set logical-systems B protocols bgp group external-peers family inet6 unicast
set logical-systems B routing-options router-id 172.16.2.2
set logical-systems B routing-options autonomous-system 22
```

Device C

```
set logical-systems C interfaces lt-0/1/0 unit 10 description to-E
set logical-systems C interfaces lt-0/1/0 unit 10 encapsulation frame-relay
set logical-systems C interfaces lt-0/1/0 unit 10 dlci 10
set logical-systems C interfaces lt-0/1/0 unit 10 peer-unit 9
set logical-systems C interfaces lt-0/1/0 unit 10 family inet6 address 2001:db8:0:3::/64 eui-64
set logical-systems C interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set logical-systems C protocols bgp group external-peers type external
set logical-systems C protocols bgp group external-peers peer-as 17
set logical-systems C protocols bgp group external-peers neighbor 2001:db8:0:3:2a0:a502:0:9da
set logical-systems C protocols bgp group external-peers family inet6 unicast
set logical-systems C routing-options router-id 172.16.3.3
set logical-systems C routing-options autonomous-system 22
```

Device D

```
set logical-systems D interfaces lt-0/1/0 unit 7 description to-E
set logical-systems D interfaces lt-0/1/0 unit 7 encapsulation frame-relay
set logical-systems D interfaces lt-0/1/0 unit 7 dlci 7
set logical-systems D interfaces lt-0/1/0 unit 7 peer-unit 21
set logical-systems D interfaces lt-0/1/0 unit 7 family inet6 address 2001:db8:0:4::/64 eui-64
set logical-systems D interfaces lo0 unit 4 family inet6 address 2001:db8::4/128
set logical-systems D protocols bgp group external-peers type external
set logical-systems D protocols bgp group external-peers peer-as 17
set logical-systems D protocols bgp group external-peers neighbor 2001:db8:0:4:2a0:a502:0:15da
set logical-systems D protocols bgp group external-peers family inet6 unicast
set logical-systems D routing-options router-id 172.16.4.4
set logical-systems D routing-options autonomous-system 79
```

Device E

```
set logical-systems E interfaces lt-0/1/0 unit 5 description to-B
set logical-systems E interfaces lt-0/1/0 unit 5 encapsulation frame-relay
```

```
set logical-systems E interfaces lt-0/1/0 unit 5 dlci 6
set logical-systems E interfaces lt-0/1/0 unit 5 peer-unit 6
set logical-systems E interfaces lt-0/1/0 unit 5 family inet6 address 2001:db8:0:2::/64 eui-64
set logical-systems E interfaces lt-0/1/0 unit 9 description to-C
set logical-systems E interfaces lt-0/1/0 unit 9 encapsulation frame-relay
set logical-systems E interfaces lt-0/1/0 unit 9 dlci 10
set logical-systems E interfaces lt-0/1/0 unit 9 peer-unit 10
set logical-systems E interfaces lt-0/1/0 unit 9 family inet6 address 2001:db8:0:3::/64 eui-64
set logical-systems E interfaces lt-0/1/0 unit 21 description to-D
set logical-systems E interfaces lt-0/1/0 unit 21 encapsulation frame-relay
set logical-systems E interfaces lt-0/1/0 unit 21 dlci 7
set logical-systems E interfaces lt-0/1/0 unit 21 peer-unit 7
set logical-systems E interfaces lt-0/1/0 unit 21 family inet6 address 2001:db8:0:4::/64 eui-64
set logical-systems E interfaces lt-0/1/0 unit 25 description to-A
set logical-systems E interfaces lt-0/1/0 unit 25 encapsulation frame-relay
set logical-systems E interfaces lt-0/1/0 unit 25 dlci 1
set logical-systems E interfaces lt-0/1/0 unit 25 peer-unit 1
set logical-systems E interfaces lt-0/1/0 unit 25 family inet6 address 2001:db8:0:1::/64 eui-64
set logical-systems E interfaces lo0 unit 5 family inet6 address 2001:db8::5/128
set logical-systems E protocols bgp group external-peers type external
set logical-systems E protocols bgp group external-peers peer-as 22
set logical-systems E protocols bgp group external-peers neighbor 2001:db8:0:1:2a0:a502:0:1da
set logical-systems E protocols bgp group external-peers neighbor 2001:db8:0:2:2a0:a502:0:6da
set logical-systems E protocols bgp group external-peers neighbor 2001:db8:0:3:2a0:a502:0:ada
set logical-systems E protocols bgp group external-peers neighbor 2001:db8:0:4:2a0:a502:0:7da
peer-as 79
set logical-systems E protocols bgp group external-peers family inet6 unicast
set logical-systems E routing-options router-id 172.16.5.5
set logical-systems E routing-options autonomous-system 17
```

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 the BGP peer sessions:

1. Run the `show interfaces terse` command to verify that the physical router has a logical tunnel (**lt**) interface.

```
user@R1> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
...
lt-0/1/0                up    up
...
```

2. On Logical System A, configure the interface encapsulation, peer-unit number, and DLCI to reach Logical System E.

```
user@R1> set cli logical-system A
Logical system: A
[edit]
user@R1:A> edit
Entering configuration mode
[edit]
user@R1:A# edit interfaces
[edit interfaces]
user@R1:A# set lt-0/1/0 unit 1 encapsulation frame-relay
user@R1:A# set lt-0/1/0 unit 1 dlci 1
user@R1:A# set lt-0/1/0 unit 1 peer-unit 25
```

3. On Logical System A, configure the network address for the link to Peer E, and configure a loopback interface.

```
[edit interfaces]
user@R1:A# set lt-0/1/0 unit 1 description to-E
user@R1:A# set lt-0/1/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
user@R1:A# set lo0 unit 1 family inet6 address 2001:db8::1/128
```

4. On Logical System E, configure the interface encapsulation, peer-unit number, and DLCI to reach Logical System A.

```
user@R1> set cli logical-system E
Logical system: E
[edit]
user@R1:E> edit
```

```
Entering configuration mode
[edit]
user@R1:E# edit interfaces
[edit interfaces]
user@R1:E# set lt-0/1/0 unit 25 encapsulation frame-relay
user@R1:E# set lt-0/1/0 unit 25 dlci 1
user@R1:E# set lt-0/1/0 unit 25 peer-unit 1
```

5. On Logical System E, configure the network address for the link to Peer A, and configure a loopback interface.

```
[edit interfaces]
user@R1:E# set lt-0/1/0 unit 25 description to-A
user@R1:E# set lt-0/1/0 unit 25 family inet6 address 2001:db8:0:1::/64 eui-64
user@R1:E# set lo0 unit 5 family inet6 address 2001:db8::5/128
```

6. Run the `show interfaces terse` command to see the IPv6 addresses that are generated by EUI-64.

   The 2001 addresses are used in this example in the BGP `neighbor` statements.

NOTE: The fe80 addresses are link-local addresses and are not used in this example.

```
user@R1:A> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
Logical system: A

betsy@tp8:A> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
lt-0/1/0
lt-0/1/0.1              up    up   inet6    2001:db8:0:1:2a0:a502:0:1da/64
                                            fe80::2a0:a502:0:1da/64
lo0
lo0.1                   up    up   inet6    2001:db8::1
                                            fe80::2a0:a50f:fc56:1da
```

```
user@R1:E> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
lt-0/1/0
```

```
lt-0/1/0.25            up    up   inet6    2001:db8:0:1:2a0:a502:0:19da/64
                                           fe80::2a0:a502:0:19da/64
lo0
lo0.5                  up    up   inet6    2001:db8::5
                                           fe80::2a0:a50f:fc56:1da
```

7. Repeat the interface configuration on the other logical systems.

Configuring the External BGP Sessions

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 the BGP peer sessions:

1. On Logical System A, create the BGP group, and add the external neighbor address.

```
[edit protocols bgp group external-peers]
user@R1:A# set neighbor 2001:db8:0:1:2a0:a502:0:19da
user@R1:A# set family inet6 unicast
```

2. On Logical System E, create the BGP group, and add the external neighbor address.

```
[edit protocols bgp group external-peers]
user@R1:E# set neighbor 2001:db8:0:1:2a0:a502:0:1da
user@R1:E# set family inet6 unicast
```

3. On Logical System A, specify the autonomous system (AS) number of the external AS.

```
[edit protocols bgp group external-peers]
user@R1:A# set peer-as 17
```

4. On Logical System E, specify the autonomous system (AS) number of the external AS.

```
[edit protocols bgp group external-peers]
user@R1:E# set peer-as 22
```

5. On Logical System A, set the peer type to EBGP.

```
[edit protocols bgp group external-peers]
user@R1:A# set type external
```

6. On Logical System E, set the peer type to EBGP.

```
[edit protocols bgp group external-peers]
user@R1:E# set type external
```

7. On Logical System A, set the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R1:A# set router-id 172.16.1.1
user@R1:A# set autonomous-system 22
```

8. On Logical System E, set the AS number and router ID.

```
[edit routing-options]
user@R1:E# set router-id 172.16.5.5
user@R1:E# set autonomous-system 17
```

9. Repeat these steps for Peers A, B, C, and D.

Results

From configuration mode, confirm your configuration by entering the `show logical-systems` command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R1# show logical-systems
A {
    interfaces {
        lt-0/1/0 {
            unit 1 {
                description to-E;
                encapsulation frame-relay;
```

```
                dlci 1;
                peer-unit 25;
                family inet6 {
                    address 2001:db8:0:1::/64 {
                        eui-64;
                    }
                }
            }
        }
        lo0 {
            unit 1 {
                family inet6 {
                    address 2001:db8::1/128;
                }
            }
        }
    }
    protocols {
        bgp {
            group external-peers {
                type external;
                peer-as 17;
                neighbor 2001:db8:0:1:2a0:a502:0:19da;
            }
        }
        routing-options {
            router-id 172.16.1.1;
            autonomous-system 22;
        }
    }
    B {
        interfaces {
            lt-0/1/0 {
                unit 6 {
                    description to-E;
                    encapsulation frame-relay;
                    dlci 6;
                    peer-unit 5;
                    family inet6 {
                        address 2001:db8:0:2::/64 {
                            eui-64;
                        }
                    }
```

```
            }
        }
        lo0 {
            unit 2 {
                family inet6 {
                    address 2001:db8::2/128;
                }
            }
        }
    }
    protocols {
        bgp {
            group external-peers {
                type external;
                peer-as 17;
                neighbor 2001:db8:0:2:2a0:a502:0:5da;
            }
        }
        routing-options {
            router-id 172.16.2.2;
            autonomous-system 22;
        }
    }
    C {
        interfaces {
            lt-0/1/0 {
                unit 10 {
                    description to-E;
                    encapsulation frame-relay;
                    dlci 10;
                    peer-unit 9;
                    family inet6 {
                        address 2001:db8:0:3::/64 {
                            eui-64;
                        }
                    }
                }
            }
            lo0 {
                unit 3 {
                    family inet6 {
                        address 2001:db8::3/128;
                    }
```

```
                }
            }
        }
        protocols {
            bgp {
                group external-peers {
                    type external;
                    peer-as 17;
                    neighbor 2001:db8:0:3:2a0:a502:0:9da;
                }
            }
        }
        routing-options {
            router-id 172.16.3.3;
            autonomous-system 22;
        }
    }
    D {
        interfaces {
            lt-0/1/0 {
                unit 7 {
                    description to-E;
                    encapsulation frame-relay;
                    dlci 7;
                    peer-unit 21;
                    family inet6 {
                        address 2001:db8:0:4::/64 {
                            eui-64;
                        }
                    }
                }
            }
            lo0 {
                unit 4 {
                    family inet6 {
                        address 2001:db8::4/128;
                    }
                }
            }
        }
        protocols {
            bgp {
                group external-peers {
```

```
                type external;
                peer-as 17;
                neighbor 2001:db8:0:4:2a0:a502:0:15da;
            }
        }
        routing-options {
            router-id 172.16.4.4;
            autonomous-system 79;
        }
    }
    E {
        interfaces {
            lt-0/1/0 {
                unit 5 {
                    description to-B;
                    encapsulation frame-relay;
                    dlci 6;
                    peer-unit 6;
                    family inet6 {
                        address 2001:db8:0:2::/64 {
                            eui-64;
                        }
                    }
                }
                unit 9 {
                    description to-C;
                    encapsulation frame-relay;
                    dlci 10;
                    peer-unit 10;
                    family inet6 {
                        address 2001:db8:0:3::/64 {
                            eui-64;
                        }
                    }
                }
                unit 21 {
                    description to-D;
                    encapsulation frame-relay;
                    dlci 7;
                    peer-unit 7;
                    family inet6 {
                        address 2001:db8:0:4::/64 {
                            eui-64;
```

```
                        }
                    }
                }
                unit 25 {
                    description to-A;
                    encapsulation frame-relay;
                    dlci 1;
                    peer-unit 1;
                    family inet6 {
                        address 2001:db8:0:1::/64 {
                            eui-64;
                        }
                    }
                }
            }
            lo0 {
                unit 5 {
                    family inet6 {
                        address 2001:db8::5/128;
                    }
                }
            }
        }
        protocols {
            bgp {
                group external-peers {
                    type external;
                    peer-as 22;
                    neighbor 2001:db8:0:1:2a0:a502:0:1da;
                    neighbor 2001:db8:0:2:2a0:a502:0:6da;
                    neighbor 2001:db8:0:3:2a0:a502:0:ada;
                    neighbor 2001:db8:0:4:2a0:a502:0:7da {
                        peer-as 79;
                    }
                }
            }
        }
        routing-options {
            router-id 172.16.5.5;
            autonomous-system 17;
        }
    }
```

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

Verification

IN THIS SECTION

- Verifying BGP Neighbors | **54**
- Verifying BGP Groups | **58**
- Verifying BGP Summary Information | **58**
- Checking the Routing Table | **59**

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is active for each neighbor address.

Action

From operational mode, run the `show bgp neighbor` command.

```
user@R1:E> show bgp neighbor
Peer: 2001:db8:0:1:2a0:a502:0:1da+54987 AS 22 Local: 2001:db8:0:1:2a0:a502:0:19da+179 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: Open Message Error
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Error: 'Open Message Error' Sent: 20 Recv: 0
  Peer ID: 172.16.1.1         Local ID: 172.16.5.5          Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: lt-0/1/0.25
  NLRI for restart configured on peer: inet6-unicast
```

```
  NLRI advertised by peer: inet6-unicast
  NLRI for this session: inet6-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: inet6-unicast
  NLRI of received end-of-rib markers: inet6-unicast
  NLRI of all end-of-rib markers sent: inet6-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet6.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 7    Sent 18   Checked 81
  Input messages:  Total 1611   Updates 1       Refreshes 0     Octets 30660
  Output messages: Total 1594   Updates 0       Refreshes 0     Octets 30356
  Output Queue[0]: 0

Peer: 2001:db8:0:2:2a0:a502:0:6da+179 AS 22 Local: 2001:db8:0:2:2a0:a502:0:5da+55502 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: Open Message Error
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Error: 'Open Message Error' Sent: 26 Recv: 0
  Peer ID: 172.16.2.2         Local ID: 172.16.5.5           Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 2
  BFD: disabled, down
  Local Interface: lt-0/1/0.5
  NLRI for restart configured on peer: inet6-unicast
  NLRI advertised by peer: inet6-unicast
  NLRI for this session: inet6-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: inet6-unicast
  NLRI of received end-of-rib markers: inet6-unicast
```

```
  NLRI of all end-of-rib markers sent: inet6-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet6.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 15   Sent 8    Checked 8
  Input messages:  Total 1610   Updates 1       Refreshes 0     Octets 30601
  Output messages: Total 1645   Updates 0       Refreshes 0     Octets 32417
  Output Queue[0]: 0

Peer: 2001:db8:0:3:2a0:a502:0:ada+55983 AS 22 Local: 2001:db8:0:3:2a0:a502:0:9da+179 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 172.16.3.3         Local ID: 172.16.5.5           Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 3
  BFD: disabled, down
  Local Interface: lt-0/1/0.9
  NLRI for restart configured on peer: inet6-unicast
  NLRI advertised by peer: inet6-unicast
  NLRI for this session: inet6-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: inet6-unicast
  NLRI of received end-of-rib markers: inet6-unicast
  NLRI of all end-of-rib markers sent: inet6-unicast
  Peer supports 4 byte AS extension (peer-as 22)
  Peer does not support Addpath
  Table inet6.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
```

```
    Accepted prefixes:              0
    Suppressed due to damping:      0
    Advertised prefixes:            0
  Last traffic (seconds): Received 21   Sent 21   Checked 67
  Input messages:  Total 1610   Updates 1       Refreshes 0     Octets 30641
  Output messages: Total 1587   Updates 0       Refreshes 0     Octets 30223
  Output Queue[0]: 0

Peer: 2001:db8:0:4:2a0:a502:0:7da+49255 AS 79 Local: 2001:db8:0:4:2a0:a502:0:15da+179 AS 17
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 172.16.4.4         Local ID: 172.16.5.5           Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  Local Interface: lt-0/1/0.21
  NLRI for restart configured on peer: inet6-unicast
  NLRI advertised by peer: inet6-unicast
  NLRI for this session: inet6-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: inet6-unicast
  NLRI of received end-of-rib markers: inet6-unicast
  NLRI of all end-of-rib markers sent: inet6-unicast
  Peer supports 4 byte AS extension (peer-as 79)
  Peer does not support Addpath
  Table inet6.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 6    Sent 17   Checked 25
  Input messages:  Total 1615   Updates 1       Refreshes 0     Octets 30736
  Output messages: Total 1593   Updates 0       Refreshes 0     Octets 30337
  Output Queue[0]: 0
```

Meaning

IPv6 unicast network layer reachability information (NLRI) is being exchanged between the neighbors.

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From operational mode, run the `show bgp group` command.

```
user@R1:E> show bgp group
Group Type: External                              Local AS: 17
  Name: external-peers  Index: 0                  Flags: <>
  Holdtime: 0
  Total peers: 4        Established: 4
  2001:db8:0:1:2a0:a502:0:1da+54987
  2001:db8:0:2:2a0:a502:0:6da+179
  2001:db8:0:3:2a0:a502:0:ada+55983
  2001:db8:0:4:2a0:a502:0:7da+49255
  inet6.0: 0/0/0/0

Groups: 1  Peers: 4    External: 4    Internal: 0    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0                0          0          0          0          0          0
inet6.2                0          0          0          0          0          0
```

Meaning

The group type is external, and the group has four peers.

Verifying BGP Summary Information

Purpose

Verify that the BGP peer relationships are established.

Action

From operational mode, run the `show bgp summary` command.

```
user@R1:E> show bgp summary
Groups: 1 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0                0          0          0          0          0          0
inet6.2                0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
2001:db8:0:1:2a0:a502:0:1da          22       1617       1600       0       0    12:07:00 Establ
  inet6.0: 0/0/0/0
2001:db8:0:2:2a0:a502:0:6da          22       1616       1651       0       0    12:06:56 Establ
  inet6.0: 0/0/0/0
2001:db8:0:3:2a0:a502:0:ada          22       1617       1594       0       0    12:04:32 Establ
  inet6.0: 0/0/0/0
2001:db8:0:4:2a0:a502:0:7da          79       1621       1599       0       0    12:07:00 Establ
  inet6.0: 0/0/0/0
```

Meaning

The Down peers: 0 output shows that the BGP peers are in the established state.

Checking the Routing Table

Purpose

Verify that the inet6.0 routing table is populated with local and direct routes.

Action

From operational mode, run the `show route` command.

```
user@R1:E> show route
inet6.0: 15 destinations, 18 routes (15 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::5/128    *[Direct/0] 12:41:18
                    > via lo0.5
```

```
2001:db8:0:1::/64  *[Direct/0] 14:40:01
                    > via lt-0/1/0.25
2001:db8:0:1:2a0:a502:0:19da/128
                   *[Local/0] 14:40:01
                      Local via lt-0/1/0.25
2001:db8:0:2::/64  *[Direct/0] 14:40:02
                    > via lt-0/1/0.5
2001:db8:0:2:2a0:a502:0:5da/128
                   *[Local/0] 14:40:02
                      Local via lt-0/1/0.5
2001:db8:0:3::/64  *[Direct/0] 14:40:02
                    > via lt-0/1/0.9
2001:db8:0:3:2a0:a502:0:9da/128
                   *[Local/0] 14:40:02
                      Local via lt-0/1/0.9
2001:db8:0:4::/64  *[Direct/0] 14:40:01
                    > via lt-0/1/0.21
2001:db8:0:4:2a0:a502:0:15da/128
                   *[Local/0] 14:40:01
                      Local via lt-0/1/0.21
fe80::/64          *[Direct/0] 14:40:02
                    > via lt-0/1/0.5
                    [Direct/0] 14:40:02
                    > via lt-0/1/0.9
                    [Direct/0] 14:40:01
                    > via lt-0/1/0.21
                    [Direct/0] 14:40:01
                    > via lt-0/1/0.25
fe80::2a0:a502:0:5da/128
                   *[Local/0] 14:40:02
                      Local via lt-0/1/0.5
fe80::2a0:a502:0:9da/128
                   *[Local/0] 14:40:02
                      Local via lt-0/1/0.9
fe80::2a0:a502:0:15da/128
                   *[Local/0] 14:40:01
                      Local via lt-0/1/0.21
fe80::2a0:a502:0:19da/128
                   *[Local/0] 14:40:01
                      Local via lt-0/1/0.25
fe80::2a0:a50f:fc56:1da/128
```

```
                    *[Direct/0] 12:41:18
                     > via lo0.5
```

Meaning

The inet6.0 routing table contains local and direct routes. To populate the routing table with other types of routes, you must configure routing policies.

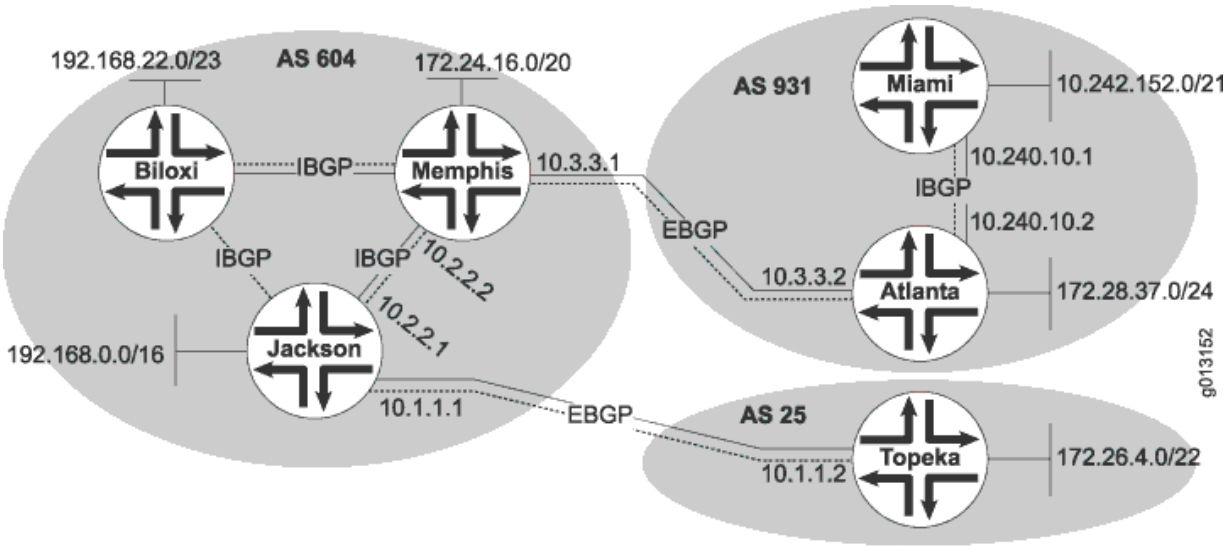
SEE ALSO

- Understanding External BGP Peering Sessions

Understanding Internal BGP Peering Sessions

When two BGP-enabled devices are in the same autonomous system (AS), the BGP session is called an *internal* BGP session, or IBGP session. BGP uses the same message types on IBGP and external BGP (EBGP) sessions, but the rules for when to send each message and how to interpret each message differ slightly. For this reason, some people refer to IBGP and EBGP as two separate protocols.

Figure 5: Internal and External BGP



In Figure 5 on page 61, Device Jackson, Device Memphis, and Device Biloxi have IBGP peer sessions with each other. Likewise, Device Miami and Device Atlanta have IBGP peer sessions between each other.

The purpose of IBGP is to provide a means by which EBGP route advertisements can be forwarded throughout the network. In theory, to accomplish this task you could redistribute all of your EBGP routes into an interior gateway protocol (IGP), such as OSPF or IS-IS. This, however, is not recommended in a production environment because of the large number of EBGP routes in the Internet and because of the way that IGPs operate. In short, with that many routes the IGP churns or crashes.

Generally, the loopback interface (lo0) is used to establish connections between IBGP peers. The loopback interface is always up as long as the device is operating. If there is a route to the loopback address, the IBGP peering session stays up. If a physical interface address is used instead and that interface goes up and down, the IBGP peering session also goes up and down. Thus the loopback interface provides fault tolerance in case the physical interface or the link goes down, if the device has link redundancy.

While IBGP neighbors do not need to be directly connected, they do need to be fully meshed. In this case, fully meshed means that each device is logically connected to every other device through neighbor peer relationships. The `neighbor` statement creates the mesh. Because of the full mesh requirement of IBGP, you must configure individual peering sessions between all IBGP devices in the AS. The full mesh need not be physical links. Rather, the configuration on each routing device must create a full mesh of peer sessions (using multiple `neighbor` statements).

NOTE: The requirement for a full mesh is waived if you configure a confederation or route reflection.

To understand the full-mesh requirement, consider that an IBGP-learned route cannot be readvertised to another IBGP peer. The reason for preventing the readvertisement of IBGP routes and requiring the full mesh is to avoid routing loops within an AS. The AS path attribute is the means by which BGP routing devices avoid loops. The path information is examined for the local AS number only when the route is received from an EBGP peer. Because the attribute is only modified across AS boundaries, this system works well. However, the fact that the attribute is only modified across AS boundaries presents an issue inside the AS. For example, suppose that routing devices A, B, and C are all in the same AS. Device A receives a route from an EBGP peer and sends the route to Device B, which installs it as the active route. The route is then sent to Device C, which installs it locally and sends it back to Device A. If Device A installs the route, a loop is formed within the AS. The routing devices are not able to detect the loop because the AS path attribute is not modified during these advertisements. Therefore, the BGP protocol designers decided that the only assurance of never forming a routing loop was to prevent an IBGP peer from advertising an IBGP-learned route within the AS. For route reachability, the IBGP peers are fully meshed.

IBGP supports multihop connections, so IBGP neighbors can be located anywhere within the AS and often do not share a link. A recursive route lookup resolves the loopback peering address to an IP

forwarding next hop. The lookup service is provided by static routes or an IGP such as OSPF, or BGP routes.

SEE ALSO

Example: Configuring Internal BGP Peering Sessions on Logical Systems | 80

Example: Configuring Internal BGP Peer Sessions

IN THIS SECTION

- Requirements | 63
- Overview | 63
- Configuration | 65
- Verification | 76

This example shows how to configure internal BGP peer sessions.

Requirements

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

Overview

In this example, you configure internal BGP (IBGP) peer sessions. The loopback interface (lo0) is used to establish connections between IBGP peers. The loopback interface is always up as long as the device is operating. If there is a route to the loopback address, the IBGP peer session stays up. If a physical interface address is used instead and that interface goes up and down, the IBGP peer session also goes up and down. Thus, if the device has link redundancy, the loopback interface provides fault tolerance in case the physical interface or one of the links goes down.

When a device peers with a remote device's loopback interface address, the local device expects BGP update messages to come from (be sourced by) the remote device's loopback interface address. The `local-address` statement enables you to specify the source information in BGP update messages. If you omit the `local-address` statement, the expected source of BGP update messages is based on the device's source address selection rules, which normally results in the egress interface address being the expected source of update messages. When this happens, the peer session is not established because a mismatch

exists between the expected source address (the egress interface of the peer) and the actual source (the loopback interface of the peer). To make sure that the expected source address matches the actual source address, specify the loopback interface address in the `local-address` statement.

Because IBGP supports multihop connections, IBGP neighbors can be located anywhere within the autonomous system (AS) and often do not share a link. A recursive route lookup resolves the loopback peer address to an IP forwarding next hop. In this example, this service is provided by OSPF. Although interior gateway protocol (IGP) neighbors do not need to be directly connected, they do need to be fully meshed. In this case, fully meshed means that each device is logically connected to every other device through neighbor peer relationships. The `neighbor` statement creates the mesh.

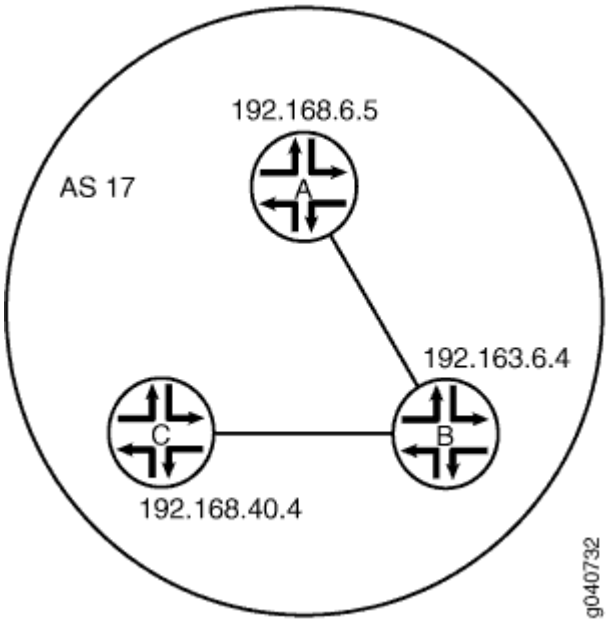
NOTE: The requirement for a full mesh is waived if you configure a confederation or route reflection.

After the BGP peers are established, local routes are not automatically advertised by the BGP peers. At each BGP-enabled device, policy configuration is required to export the local, static, or IGP-learned routes into the BGP routing information base (RIB) and then advertise them as BGP routes to the other peers. BGP's advertisement policy, by default, does not advertise any non-BGP routes (such as local routes) to peers.

In the sample network, the devices in AS 17 are fully meshed in the group **internal-peers**. The devices have loopback addresses 192.168.6.5, 192.163.6.4, and 192.168.40.4.

Figure 6 on page 65 shows a typical network with internal peer sessions.

Figure 6: Typical Network with IBGP Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | 65
- Configuring Device A | 67
- Configuring Device B | 70
- Configuring Device C | 73

CLI Quick Configuration

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

Device A

```
set interfaces ge-0/1/0 unit 1 description to-B
set interfaces ge-0/1/0 unit 1 family inet address 10.10.10.1/30
set interfaces lo0 unit 1 family inet address 192.168.6.5/32
```

```
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers description “connections to B and C”
set protocols bgp group internal-peers local-address 192.168.6.5
set protocols bgp group internal-peers export send-direct
set protocols bgp group internal-peers neighbor 192.163.6.4
set protocols bgp group internal-peers neighbor 192.168.40.4
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface ge-0/1/0.1
set policy-options policy-statement send-direct term 2 from protocol direct
set policy-options policy-statement send-direct term 2 then accept
set routing-options router-id 192.168.6.5
set routing-options autonomous-system 17
```

Device B

```
set interfaces ge-0/1/0 unit 2 description to-A
set interfaces ge-0/1/0 unit 2 family inet address 10.10.10.2/30
set interfaces ge-0/1/1 unit 5 description to-C
set interfaces ge-0/1/1 unit 5 family inet address 10.10.10.5/30
set interfaces lo0 unit 2 family inet address 192.163.6.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers description “connections to A and C”
set protocols bgp group internal-peers local-address 192.163.6.4
set protocols bgp group internal-peers export send-direct
set protocols bgp group internal-peers neighbor 192.168.40.4
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface ge-0/1/0.2
set protocols ospf area 0.0.0.0 interface ge-0/1/1.5
set policy-options policy-statement send-direct term 2 from protocol direct
set policy-options policy-statement send-direct term 2 then accept
set routing-options router-id 192.163.6.4
set routing-options autonomous-system 17
```

Device C

```
set interfaces ge-0/1/0 unit 6 description to-B
set interfaces ge-0/1/0 unit 6 family inet address 10.10.10.6/30
set interfaces lo0 unit 3 family inet address 192.168.40.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers description “connections to A and B”
```

```
set protocols bgp group internal-peers local-address 192.168.40.4
set protocols bgp group internal-peers export send-direct
set protocols bgp group internal-peers neighbor 192.163.6.4
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface ge-0/1/0.6
set policy-options policy-statement send-direct term 2 from protocol direct
set policy-options policy-statement send-direct term 2 then accept
set routing-options router-id 192.168.40.4
set routing-options autonomous-system 17
```

Configuring Device A

Step-by-Step Procedure

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

To configure internal BGP peer sessions on Device A:

1. Configure the interfaces.

```
[edit interfaces ge-0/1/0 unit 1]
user@A# set description to-B
user@A# set family inet address 10.10.10.1/30
[edit interfaces]
user@A# set lo0 unit 1 family inet address 192.168.6.5/32
```

2. Configure BGP.

   The `neighbor` statements are included for both Device B and Device C, even though Device A is not directly connected to Device C.

```
[edit protocols bgp group internal-peers]
user@A# set type internal
user@A# set description “connections to B and C”
user@A# set local-address 192.168.6.5
user@A# set export send-direct
user@A# set neighbor 192.163.6.4
user@A# set neighbor 192.168.40.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@A# set interface lo0.1 passive
user@A# set interface ge-0/1/0.1
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 2]
user@A# set from protocol direct
user@A# set then accept
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@A# set router-id 192.168.6.5
user@A# set autonomous-system 17
```

Results

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

```
user@A# show interfaces
ge-0/1/0 {
    unit 1 {
        description to-B;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
```

```
            address 192.168.6.5/32;
        }
    }
}
```

```
user@A# show policy-options
policy-statement send-direct {
    term 2 {
        from protocol direct;
        then accept;
    }
}
```

```
user@A# show protocols
bgp {
    group internal-peers {
        type internal;
        description “connections to B and C”;
        local-address 192.168.6.5;
        export send-direct;
        neighbor 192.163.6.4;
        neighbor 192.168.40.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface ge-0/1/0.1;
    }
}
```

```
user@A# show routing-options
router-id 192.168.6.5;
autonomous-system 17;
```

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

Configuring Device B

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*.

To configure internal BGP peer sessions on Device B:

1. Configure the interfaces.

```
[edit interfaces ge-0/1/0 unit 2]
user@B# set description to-A
user@B# set family inet address 10.10.10.2/30
[edit interfaces ge-0/1/1]
user@B# set unit 5 description to-C
user@B# set unit 5 family inet address 10.10.10.5/30
[edit interfaces]
user@B# set lo0 unit 2 family inet address 192.163.6.4/32
```

2. Configure BGP.

   The `neighbor` statements are included for both Device B and Device C, even though Device A is not directly connected to Device C.

```
[edit protocols bgp group internal-peers]
user@B# set type internal
user@B# set description “connections to A and C”
user@B# set local-address 192.163.6.4
user@B# set export send-direct
user@B# set neighbor 192.168.40.4
user@B# set neighbor 192.168.6.5
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@B# set interface lo0.2 passive
user@B# set interface ge-0/1/0.2
user@B# set interface ge-0/1/1.5
```

4. Configure a policy that accepts direct routes.

Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 2]
user@B# set from protocol direct
user@B# set then accept
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@B# set router-id 192.163.6.4
user@B# set autonomous-system 17
```

Results

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

```
user@B# show interfaces
ge-0/1/0 {
    unit 2 {
        description to-A;
        family inet {
            address 10.10.10.2/30;
        }
    }
}
ge-0/1/1 {
    unit 5 {
        description to-C;
        family inet {
            address 10.10.10.5/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
```

```
            address 192.163.6.4/32;
        }
    }
}
```

```
user@B# show policy-options
policy-statement send-direct {
    term 2 {
        from protocol direct;
        then accept;
    }
}
```

```
user@B# show protocols
bgp {
    group internal-peers {
        type internal;
        description “connections to A and C”;
        local-address 192.163.6.4;
        export send-direct;
        neighbor 192.168.40.4;
        neighbor 192.168.6.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface ge-0/1/0.2;
        interface ge-0/1/1.5;
    }
}
```

```
user@B# show routing-options
router-id 192.163.6.4;
autonomous-system 17;
```

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

Configuring Device C

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 Junos OS CLI User Guide.

To configure internal BGP peer sessions on Device C:

1. Configure the interfaces.

```
[edit interfaces ge-0/1/0 unit 6]
user@C# set description to-B
user@C# set family inet address 10.10.10.6/30
[edit interfaces]
user@C# set lo0 unit 3 family inet address 192.168.40.4/32
```

2. Configure BGP.

   The `neighbor` statements are included for both Device B and Device C, even though Device A is not directly connected to Device C.

```
[edit protocols bgp group internal-peers]
user@C# set type internal
user@C# set description “connections to A and B”
user@C# set local-address 192.168.40.4
user@C# set export send-direct
user@C# set neighbor 192.163.6.4
user@C# set neighbor 192.168.6.5
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@C# set interface lo0.3 passive
user@C# set interface ge-0/1/0.6
```

4. Configure a policy that accepts direct routes.

Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 2]
user@C# set from protocol direct
user@C# set then accept
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@C# set router-id 192.168.40.4
user@C# set autonomous-system 17
```

Results

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

```
user@C# show interfaces
ge-0/1/0 {
    unit 6 {
        description to-B;


        family inet {
            address 10.10.10.6/30;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.40.4/32;
        }
```

```
    }
}
```

```
user@C# show policy-options
policy-statement send-direct {
    term 2 {
        from protocol direct;
        then accept;
    }
}
```

```
user@C# show protocols
bgp {
    group internal-peers {
        type internal;
        description “connections to A and B”;
        local-address 192.168.40.4;
        export send-direct;
        neighbor 192.163.6.4;
        neighbor 192.168.6.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.3 {
            passive;
        }
        interface ge-0/1/0.6;
    }
}
```

```
user@C# show routing-options
router-id 192.168.40.4;
autonomous-system 17;
```

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

Verification

IN THIS SECTION

- Verifying BGP Neighbors | 76
- Verifying BGP Groups | 78
- Verifying BGP Summary Information | 79
- Verifying That BGP Routes Are Installed in the Routing Table | 79

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is active for each neighbor address.

Action

From operational mode, enter the `show bgp neighbor` command.

```
user@A> show bgp neighbor
Peer: 192.163.6.4+179 AS 17    Local: 192.168.6.5+58852 AS 17
  Type: Internal    State: Established    Flags: Sync
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct ]
  Options: Preference LocalAddress Refresh
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.163.6.4     Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
```

```
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            3
    Accepted prefixes:            3
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 25   Sent 19   Checked 67
  Input messages:  Total 2420   Updates 4       Refreshes 0     Octets 46055
  Output messages: Total 2411   Updates 2       Refreshes 0     Octets 45921
  Output Queue[0]: 0

Peer: 192.168.40.4+179 AS 17   Local: 192.168.6.5+56466 AS 17
  Type: Internal    State: Established    Flags: Sync
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct ]
  Options: Preference LocalAddress Refresh
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.40.4    Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
```

```
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            2
    Accepted prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 7    Sent 21   Checked 24
  Input messages:  Total 2412   Updates 2       Refreshes 0     Octets 45867
  Output messages: Total 2409   Updates 2       Refreshes 0     Octets 45883
  Output Queue[0]: 0
```

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From operational mode, enter the `show bgp group` command.

```
user@A> show bgp group
Group Type: Internal    AS: 17                    Local AS: 17
  Name: internal-peers  Index: 0                  Flags: <Export Eval>
  Export: [ send-direct ]
  Holdtime: 0
  Total peers: 2        Established: 2
  192.163.6.4+179
  192.168.40.4+179
  inet.0: 0/5/5/0

Groups: 1  Peers: 2    External: 0    Internal: 2    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 5          0          0          0          0          0
```

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From operational mode, enter the `show bgp summary` command.

```
user@A> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 5          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.163.6.4              17       2441       2432       0       0    18:18:52
0/3/3/0              0/0/0/0
192.168.40.4             17       2432       2430       0       0    18:18:48
0/2/2/0              0/0/0/0
```

Verifying That BGP Routes Are Installed in the Routing Table

Purpose

Verify that the export policy configuration is causing the BGP routes to be installed in the routing tables of the peers.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@A> show route protocol bgp
inet.0: 7 destinations, 12 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.0/30       [BGP/170] 07:09:57, localpref 100, from 192.163.6.4
                      AS path: I
                    > to 10.10.10.2 via ge-0/1/0.1
10.10.10.4/30       [BGP/170] 07:09:57, localpref 100, from 192.163.6.4
```

```
                    AS path: I
                  > to 10.10.10.2 via ge-0/1/0.1
                  [BGP/170] 07:07:12, localpref 100, from 192.168.40.4
                    AS path: I
                  > to 10.10.10.2 via ge-0/1/0.1
192.163.6.4/32    [BGP/170] 07:09:57, localpref 100, from 192.163.6.4
                    AS path: I
                  > to 10.10.10.2 via ge-0/1/0.1
192.168.40.4/32   [BGP/170] 07:07:12, localpref 100, from 192.168.40.4
                    AS path: I
                  > to 10.10.10.2 via ge-0/1/0.1
```

SEE ALSO

- Routing Policies, Firewall Filters, and Traffic Policers User Guide
- Understanding Internal BGP Peering Sessions | **61**
- BGP Configuration Overview | **24**

Example: Configuring Internal BGP Peering Sessions on Logical Systems

IN THIS SECTION

- Requirements | **80**
- Overview | **81**
- Configuration | **81**
- Verification | **90**

This example shows how to configure internal BGP peer sessions on logical systems.

Requirements

In this example, no special configuration beyond device initialization is required.

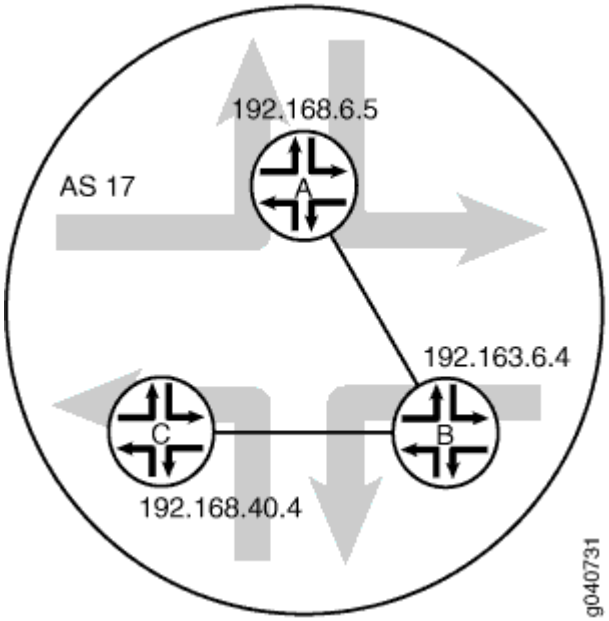
Overview

In this example, you configure internal BGP (IBGP) peering sessions.

In the sample network, the devices in AS 17 are fully meshed in the group **internal-peers**. The devices have loopback addresses 192.168.6.5, 192.163.6.4, and 192.168.40.4.

Figure 7 on page 81 shows a typical network with internal peer sessions.

Figure 7: Typical Network with IBGP Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | **82**
- Device A | **83**

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.

```
set logical-systems A interfaces lt-0/1/0 unit 1 description to-B
set logical-systems A interfaces lt-0/1/0 unit 1 encapsulation ethernet
set logical-systems A interfaces lt-0/1/0 unit 1 peer-unit 2
set logical-systems A interfaces lt-0/1/0 unit 1 family inet address 10.10.10.1/30
set logical-systems A interfaces lo0 unit 1 family inet address 192.168.6.5/32
set logical-systems A protocols bgp group internal-peers type internal
set logical-systems A protocols bgp group internal-peers local-address 192.168.6.5
set logical-systems A protocols bgp group internal-peers export send-direct
set logical-systems A protocols bgp group internal-peers neighbor 192.163.6.4
set logical-systems A protocols bgp group internal-peers neighbor 192.168.40.4
set logical-systems A protocols ospf area 0.0.0.0 interface lo0.1 passive
set logical-systems A protocols ospf area 0.0.0.0 interface lt-0/1/0.1
set logical-systems A policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems A policy-options policy-statement send-direct term 2 then accept
set logical-systems A routing-options router-id 192.168.6.5
set logical-systems A routing-options autonomous-system 17
set logical-systems B interfaces lt-0/1/0 unit 2 description to-A
set logical-systems B interfaces lt-0/1/0 unit 2 encapsulation ethernet
set logical-systems B interfaces lt-0/1/0 unit 2 peer-unit 1
set logical-systems B interfaces lt-0/1/0 unit 2 family inet address 10.10.10.2/30
set logical-systems B interfaces lt-0/1/0 unit 5 description to-C
set logical-systems B interfaces lt-0/1/0 unit 5 encapsulation ethernet
set logical-systems B interfaces lt-0/1/0 unit 5 peer-unit 6
set logical-systems B interfaces lt-0/1/0 unit 5 family inet address 10.10.10.5/30
set logical-systems B interfaces lo0 unit 2 family inet address 192.163.6.4/32
set logical-systems B protocols bgp group internal-peers type internal
set logical-systems B protocols bgp group internal-peers local-address 192.163.6.4
set logical-systems B protocols bgp group internal-peers export send-direct
set logical-systems B protocols bgp group internal-peers neighbor 192.168.40.4
set logical-systems B protocols bgp group internal-peers neighbor 192.168.6.5
set logical-systems B protocols ospf area 0.0.0.0 interface lo0.2 passive
set logical-systems B protocols ospf area 0.0.0.0 interface lt-0/1/0.2
set logical-systems B protocols ospf area 0.0.0.0 interface lt-0/1/0.5
set logical-systems B policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems B policy-options policy-statement send-direct term 2 then accept
set logical-systems B routing-options router-id 192.163.6.4
```

```
set logical-systems B routing-options autonomous-system 17
set logical-systems C interfaces lt-0/1/0 unit 6 description to-B
set logical-systems C interfaces lt-0/1/0 unit 6 encapsulation ethernet
set logical-systems C interfaces lt-0/1/0 unit 6 peer-unit 5
set logical-systems C interfaces lt-0/1/0 unit 6 family inet address 10.10.10.6/30
set logical-systems C interfaces lo0 unit 3 family inet address 192.168.40.4/32
set logical-systems C protocols bgp group internal-peers type internal
set logical-systems C protocols bgp group internal-peers local-address 192.168.40.4
set logical-systems C protocols bgp group internal-peers export send-direct
set logical-systems C protocols bgp group internal-peers neighbor 192.163.6.4
set logical-systems C protocols bgp group internal-peers neighbor 192.168.6.5
set logical-systems C protocols ospf area 0.0.0.0 interface lo0.3 passive
set logical-systems C protocols ospf area 0.0.0.0 interface lt-0/1/0.6
set logical-systems C policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems C policy-options policy-statement send-direct term 2 then accept
set logical-systems C routing-options router-id 192.168.40.4
set logical-systems C routing-options autonomous-system 17
```

Device A

Step-by-Step Procedure

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

To configure internal BGP peer sessions on Device A:

1. Configure the interfaces.

```
[edit logical-systems A interfaces lt-0/1/0 unit 1]
user@R1# set description to-B
user@R1# set encapsulation ethernet
user@R1# set peer-unit 2
user@R1# set family inet address 10.10.10.1/30
user@R1# set family inet address 192.168.6.5/32
user@R1# up
user@R1# up
[edit  logical-systems A interfaces]
user@R1# set lo0 unit 1 family inet address 192.168.6.5/32
user@R1# exit
[edit]
```

```
user@R1# edit logical-systems B interfaces lt-0/1/0
[edit logical-systems B interfaces lt-0/1/0]
user@R1# set unit 2 description to-A
user@R1# set unit 2 encapsulation ethernet
user@R1# set unit 2 peer-unit 1
user@R1# set unit 2 family inet address 10.10.10.2/30
user@R1# set unit 5 description to-C
user@R1# set unit 5 encapsulation ethernet
user@R1# set unit 5 peer-unit 6
user@R1# set family inet address 10.10.10.5/30
user@R1# up
[edit logical-systems B interfaces]
user@R1# set lo0 unit 2 family inet address 192.163.6.4/32
user@R1# exit
[edit]
user@R1# edit logical-systems C interfaces lt-0/1/0 unit 6
[edit logical-systems C interfaces lt-0/1/0 unit 6]
set description to-B
set encapsulation ethernet
set peer-unit 5
set family inet address 10.10.10.6/30
user@R1# up
user@R1# up
[edit  logical-systems C interfaces]
set lo0 unit 3 family inet address 192.168.40.4/32
```

2. Configure BGP.

   On Logical System A, the `neighbor` statements are included for both Device B and Device C, even though Logical System A is not directly connected to Device C.

```
[edit logical-systems A protocols bgp group internal-peers]
user@R1# set type internal
user@R1# set local-address 192.168.6.5
user@R1# set export send-direct
user@R1# set neighbor 192.163.6.4
user@R1# set neighbor 192.168.40.4
[edit logical-systems B protocols bgp group internal-peers]
user@R1# set type internal
user@R1# set local-address 192.163.6.4
user@R1# set export send-direct
user@R1# set neighbor 192.168.40.4
```

```
user@R1# set neighbor 192.168.6.5
[edit logical-systems C protocols bgp group internal-peers]
user@R1# set type internal
user@R1# set local-address 192.168.40.4
user@R1# set export send-direct
user@R1# set neighbor 192.163.6.4
user@R1# set neighbor 192.168.6.5
```

3. Configure OSPF.

```
[edit logical-systems A protocols ospf area 0.0.0.0]
user@R1# set interface lo0.1 passive
user@R1# set interface lt-0/1/0.1
[edit logical-systems A protocols ospf area 0.0.0.0]
user@R1# set interface lo0.2 passive
user@R1# set interface lt-0/1/0.2
user@R1# set interface lt-0/1/0.5
[edit logical-systems A protocols ospf area 0.0.0.0]
user@R1# set interface lo0.3 passive
user@R1# set interface lt-0/1/0.6
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit logical-systems A policy-options policy-statement send-direct term 2]
user@R1# set from protocol direct
user@R1# set then accept
[edit logical-systems B policy-options policy-statement send-direct term 2]
user@R1# set from protocol direct
user@R1# set then accept
[edit logical-systems C policy-options policy-statement send-direct term 2]
user@R1# set from protocol direct
user@R1# set then accept
```

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

```
[edit logical-systems A routing-options]
user@R1# set router-id 192.168.6.5
```

```
user@R1# set autonomous-system 17
[edit logical-systems B routing-options]
user@R1# set router-id 192.163.6.4
user@R1# set autonomous-system 17
[edit logical-systems C routing-options]
user@R1# set router-id 192.168.40.4
user@R1# set autonomous-system 17
```

Results

From configuration mode, confirm your configuration by entering the `show logical-systems` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@R1# show logical-systems
A {
    interfaces {
        lt-0/1/0 {
            unit 1 {
                description to-B;
                encapsulation ethernet;
                peer-unit 2;
                family inet {
                    address 10.10.10.1/30;
                }
            }
        }
        lo0 {
            unit 1 {
                family inet {
                    address 192.168.6.5/32;
                }
            }
        }
    }
    protocols {
        bgp {
            group internal-peers {
                type internal;
                local-address 192.168.6.5;
                export send-direct;
```

```
                neighbor 192.163.6.4;
                neighbor 192.168.40.4;
            }
        }
        ospf {
            area 0.0.0.0 {
                interface lo0.1 {
                    passive;
                }
                interface lt-0/1/0.1;
            }
        }
    }
    policy-options {
        policy-statement send-direct {
            term 2 {
                from protocol direct;
                then accept;
            }
        }
    }
    routing-options {
        router-id 192.168.6.5;
        autonomous-system 17;
    }
}
B {
    interfaces {
        lt-0/1/0 {
            unit 2 {
                description to-A;
                encapsulation ethernet;
                peer-unit 1;
                family inet {
                    address 10.10.10.2/30;
                }
            }
            unit 5 {
                description to-C;
                encapsulation ethernet;
                peer-unit 6;
                family inet {
                    address 10.10.10.5/30;
```

```
            }
        }
    }
    lo0 {
        unit 2 {
            family inet {
                address 192.163.6.4/32;
            }
        }
    }
}
protocols {
    bgp {
        group internal-peers {
            type internal;
            local-address 192.163.6.4;
            export send-direct;
            neighbor 192.168.40.4;
            neighbor 192.168.6.5;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.2 {
                passive;
            }
            interface lt-0/1/0.2;
            interface lt-0/1/0.5;
        }
    }
}
policy-options {
    policy-statement send-direct {
        term 2 {
            from protocol direct;
            then accept;
        }
    }
}
routing-options {
    router-id 192.163.6.4;
    autonomous-system 17;
}
```

```
}
C {
    interfaces {
        lt-0/1/0 {
            unit 6 {
                description to-B;
                encapsulation ethernet;
                peer-unit 5;
                family inet {
                    address 10.10.10.6/30;
                }
            }
        }
        lo0 {
            unit 3 {
                family inet {
                    address 192.168.40.4/32;
                }
            }
        }
    }
    protocols {
        bgp {
            group internal-peers {
                type internal;
                local-address 192.168.40.4;
                export send-direct;
                neighbor 192.163.6.4;
                neighbor 192.168.6.5;
            }
        }
        ospf {
            area 0.0.0.0 {
                interface lo0.3 {
                    passive;
                }
                interface lt-0/1/0.6;
            }
        }
    }
    policy-options {
        policy-statement send-direct {
            term 2 {
```

```
                from protocol direct;
                then accept;
            }
        }
    }
    routing-options {
        router-id 192.168.40.4;
        autonomous-system 17;
    }
}
```

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

Verification

IN THIS SECTION

- Verifying BGP Neighbors | **90**
- Verifying BGP Groups | **92**
- Verifying BGP Summary Information | **93**
- Verifying That BGP Routes Are Installed in the Routing Table | **93**

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is active for each neighbor address.

Action

From the operational mode, enter the `show bgp neighbor` command.

```
user@R1> show bgp neighbor logical-system A
Peer: 192.163.6.4+179 AS 17    Local: 192.168.6.5+58852 AS 17
  Type: Internal    State: Established    Flags: <Sync>
```

```
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct ]
  Options: <Preference LocalAddress Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.163.6.4     Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            3
    Accepted prefixes:            3
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 16   Sent 1    Checked 63
  Input messages:  Total 15713  Updates 4       Refreshes 0     Octets 298622
  Output messages: Total 15690  Updates 2       Refreshes 0     Octets 298222
  Output Queue[0]: 0

Peer: 192.168.40.4+179 AS 17   Local: 192.168.6.5+56466 AS 17
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct ]
  Options: <Preference LocalAddress Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
```

```
  Peer ID: 192.168.40.4    Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            2
    Accepted prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 15   Sent 22   Checked 68
  Input messages:  Total 15688  Updates 2       Refreshes 0     Octets 298111
  Output messages: Total 15688  Updates 2       Refreshes 0     Octets 298184
  Output Queue[0]: 0
```

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From the operational mode, enter the `show bgp group` command.

```
user@A> show bgp group logical-system A
Group Type: Internal    AS: 17                     Local AS: 17
```

```
  Name: internal-peers  Index: 0                  Flags: <Export Eval>
  Export: [ send-direct ]
  Holdtime: 0
  Total peers: 2        Established: 2
  192.163.6.4+179
  192.168.40.4+179
  inet.0: 0/5/5/0

Groups: 1  Peers: 2    External: 0    Internal: 2    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 5          0          0          0          0          0
```

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From the operational mode, enter the `show bgp summary` command.

```
user@A> show bgp summary logical-system A
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 5          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.163.6.4              17      15723      15700       0       0 4d 22:13:15
0/3/3/0              0/0/0/0
192.168.40.4             17      15698      15699       0       0 4d 22:13:11
0/2/2/0              0/0/0/0
```

Verifying That BGP Routes Are Installed in the Routing Table

Purpose

Verify that the export policy configuration is working.

Action

From the operational mode, enter the `show route protocol bgp` command.

```
user@A> show route protocol bgp logical-system A
inet.0: 7 destinations, 12 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.0/30       [BGP/170] 4d 11:05:55, localpref 100, from 192.163.6.4
                      AS path: I
                    > to 10.10.10.2 via lt-0/1/0.1
10.10.10.4/30       [BGP/170] 4d 11:05:55, localpref 100, from 192.163.6.4
                      AS path: I
                    > to 10.10.10.2 via lt-0/1/0.1
                    [BGP/170] 4d 11:03:10, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via lt-0/1/0.1
192.163.6.4/32      [BGP/170] 4d 11:05:55, localpref 100, from 192.163.6.4
                      AS path: I
                    > to 10.10.10.2 via lt-0/1/0.1
192.168.40.4/32     [BGP/170] 4d 11:03:10, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via lt-0/1/0.1
```

SEE ALSO

- Understanding Internal BGP Peering Sessions
- BGP Configuration Overview

Overview: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same Link-Local Address (IPv6)

In complex networks such as Data Center or Cloud, link-local addresses are widely used due to the high number of links and nodes. Being able to deploy multiple single-hop BGP sessions for Juniper devices using link-local addresses is a significant advantage.

Starting in Junos OS Release 20.4R1, you can enable single-hop EBGP sessions on different links over multiple directly connected peers that use the same IPv6 link-local address. You are no longer required to have unique peer addresses for Juniper devices for every EBGP session.

Example: Configure Multiple Single-Hop EBGP Sessions on Different Links Using the Same IPv6 Link-Local Address

IN THIS SECTION

- Requirements | 95
- Overview | 95
- Configuration | 96
- Verification | 100

This example shows how to configure multiple single-hop EBGP sessions on different links using the same IPv6 link-local address.

Requirements

This example uses the following hardware and software components:

- 2 MX Series routers
- Junos OS Release 20.4R1 or later version

Overview

IN THIS SECTION

- Topology | 96

Prior to Junos OS Release 20.4R1, you could configure BGP peers with link-local addresses but you could not configure multiple BGP peers to use the same link-local address on different interfaces.

Starting in Junos OS 20.4R1, you can enable multiple single-hop EBGP sessions on different links using the same link-local address.

Topology

Configuration

IN THIS SECTION

- CLI Quick Configuration | **96**
- Configure Single-Hop EBGP Sessions on Multiple Links Using the Same IPv6 Link-Local Address | **97**
- Results | **99**

In this example, you configure multiple single-hop EBGP sessions on two different links using the same IPv6 link-local address.

CLI Quick Configuration

R1

```
set interfaces ge-0/0/1
set interfaces ge-0/0/1 description R1-to-R2-Link
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet6 address fe80::10/64
set interfaces ge-0/0/1 unit 2 vlan-id 2
set interfaces ge-0/0/1 unit 2 family inet6 address fe80::10/64
set interfaces lo0 unit 0 family inet address 198.51.100.1/24 primary
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65541
```

```
set protocols bgp group external peer-as 65542
set protocols bgp group external local-as 65541
set protocols bgp group external neighbor fe80::20%ge-0/0/1.1
set protocols bgp group external neighbor "fe80::20%ge-0/0/1.2
```

R2

```
set interfaces ge-0/0/1
set interfaces ge-0/0/1 description R2-to-R1-Link
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet6 address fe80::20/64
set interfaces ge-0/0/1 unit 2 vlan-id 2
set interfaces ge-0/0/1 unit 2 family inet6 address fe80::20/64
set interfaces lo0 unit 0 family inet address 198.51.100.2/24 primary
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65542
set protocols bgp group external peer-as 65541
set protocols bgp group external local-as 65542
set protocols bgp group external neighbor fe80::10%ge-0/0/1.1
set protocols bgp group external neighbor fe80::10%ge-0/0/1.2
```

Configure Single-Hop EBGP Sessions on Multiple Links Using the Same IPv6 Link-Local Address

Step-by-Step Procedure

1. Configure basic set up, including vlan-tagging, vlan-id, loopback and IPv6 link-local addresses for R1 and R2.

   You can configure multiple units on a single interface as follows:

   R1

   ```
   set interfaces ge-0/0/1
   set interfaces ge-0/0/1 description R1-to-R2-Link
   set interfaces ge-0/0/1 vlan-tagging
   set interfaces ge-0/0/1 unit 1 vlan-id 1
   set interfaces ge-0/0/1 unit 1 family inet6 address fe80::10/64
   set interfaces ge-0/0/1 unit 2 vlan-id 2
   ```

```
set interfaces ge-0/0/1 unit 2 family inet6 address fe80::10/64
set interfaces lo0 unit 0 family inet address 198.51.100.1/24 primary
```

```
R2
set interfaces ge-0/0/1
set interfaces ge-0/0/1 description R2-to-R1-Link
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 1 vlan-id 1
set interfaces ge-0/0/1 unit 1 family inet6 address fe80::20/64
set interfaces ge-0/0/1 unit 2 vlan-id 2
set interfaces ge-0/0/1 unit 2 family inet6 address fe80::20/64
set interfaces lo0 unit 0 family inet address 198.51.100.2/24 primary
```

2. Configure routing options to enable BGP on R1 and R2.

```
R1
set routing-options router-id 198.51.100.1
set routing-options autonomous-system 65541
```

```
R2
set routing-options router-id 198.51.100.2
set routing-options autonomous-system 65542
```

3. Configure EBGP on the multiple links on R1 and R2 using the same link-local IPv6 addresses in the `set protocols bgp group` *group* `neighbor` *peeraddress%localinterface.unit* format:

```
R1
set protocols bgp group external peer-as 65542
set protocols bgp group external local-as 65541
set protocols bgp group external neighbor fe80::20%ge-0/0/1.1
set protocols bgp group external neighbor "fe80::20%ge-0/0/1.2
```

```
R2
set protocols bgp group external peer-as 65541
set protocols bgp group external local-as 65542
```

```
set protocols bgp group external neighbor fe80::10%ge-0/0/1.1
set protocols bgp group external neighbor fe80::10%ge-0/0/1.2
```

4. Enter `commit` from the configuration mode.

Results

Verify your configuration by checking the below configurations from devices as follows:

Here's how you can verify configurations on R1 device:

`user@R1# show interfaces`

```
ge-0/0/1 {
    description R1-to-R2-Link;
    vlan-tagging;
    unit 1 {
         vlan-id 1;
             family inet6 {
                address fe80::10/64;
            }
    }
    unit 2 {
         vlan-id 2;
        family inet6 {
            address fe80::10/64;
        }
    }
}
lo0 {
     unit 0 {
        family inet {
            address 198.51.100.1/24 {
                primary;
            }
        }
    }
}
```

```
user@R1# show protocols
```

```
bgp {
    group external {
        peer-as 65542;
        local-as 65541;
        neighbor "fe80::20%ge-0/0/1.1";
        neighbor "fe80::20%ge-0/0/1.2";
    }
}
```

```
user@R1# show routing-options
```

```
router-id 198.51.100.1;
autonomous-system 65541;
```

Verification

IN THIS SECTION

- Verify EBGP Link-local Support | **100**

Verify EBGP Link-local Support

Purpose

Use the `show bgp summary` command to verify the EBGP sessions created on the devices with the same link-local address through different interfaces.

Action

```
user@R1> show bgp summary
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0
```

```
                        0          0          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
fe80::20%ge-0/0/1.1       65542        115        114       0       0       50:59 Establ
  inet6.0: 0/0/0/0
fe80::20%ge-0/0/1.2       65542        114        114       0       0       50:58 Establ
  inet6.0: 0/0/0/0
```

Meaning

The output indicates that 2 EBGP sessions are established with the same IPv6 link-local address (`fe80::20`) of R2 through the 2 configured local-interfaces of R1 (`ge-0/0/1.1` and `ge-0/0/1.2`).

BGP Route Prioritization

IN THIS SECTION

- Understanding BGP Route Prioritization | **101**
- Example: Configuring the BGP Output Priority Scheduler and Global Address Family Priority | **107**
- Example: Controlling Routing Table Convergence Using BGP Route Prioritization | **115**

Understanding BGP Route Prioritization

IN THIS SECTION

- Use Cases for BGP Route Prioritization | **102**
- Properties of BGP Route Prioritization | **103**
- Understanding Queue Priority and Fairness | **105**
- Queue Servicing Procedure | **106**

While BGP is one of the most widely deployed routing protocols in use today, carrying not only network layer reachability information (NLRI) but also many types of VPN reachability information, it is notable that the protocol does not specify how the information is ordered in BGP update messages. This decision is left to the implementation.

In large-scale systems, BGP might take a significant amount of time to exchange its routing information between systems. This is especially true during BGP startup, route refresh operations, and when assisting with *graceful restart.* In order to handle the large amount of information that needs to be processed, BGP route processing is accomplished with the use of queues. Outbound routes are placed in output queues for processing. BGP route prioritization is introduced in Junos OS Release 16.1 as a means to allow the user to deterministically prioritize BGP update messages. BGP route prioritization is a process that operates strictly on the output queues., helping to order the information that is being sent to BGP peer routers.

In the default configuration, that is, when no `output-queue-priority` configuration or policy that overrides priority exists, the routing protocol process (rpd) enqueues BGP routes into the output queue per routing information base (RIB). A RIB, which is also known as a routing table, corresponds to both a specific address family, such as inet.0, and to routing instance tables such as vrf.inet.0. While processing output queues, the BGP update code flushes the output queue for the current RIB before moving on to the next RIB that has a non-empty output queue.

NOTE:

- There is no attempt to automatically prioritize routes even if there is a theoretical possibility of doing so. Prioritizing individual routes is, therefore, left completely to the user.
- If BGP route priorities are changed for a peer group, the BGP peer sessions get reset.

Use Cases for BGP Route Prioritization

Table 1 on page 103 shows the types of routes that would benefit from route prioritization and some notes about why they would benefit from it. Examples of those types of routes are also included. Prioritizing these routes within a given large-scale environment can help routers to react more quickly to important route changes.

Table 1: Use Cases for BGP Route Prioritization

Route or Update Type	Notes	Example
Prefixes used for resolving BGP next hops to an immediate forwarding next hop	Changes to these prefixes should be made as soon as possible.	• Host routes • Prefixes that are part of recursive resolution requirements
Routes used for tunnel endpoints	Tunnel endpoints such as GRE or MPLS are often used as BGP next hops.	BGP labeled unicast routes
Route types that are critical for the operation of a protocol feature	For some VPN protocols, certain route types are used to trigger time sensitive changes within the protocol. Changes to these routes must be made as soon as possible.	• MVPN Source Active Autodiscovery (Type 5) • Multihomed VPLS sites
Service provider infrastructure routes	These routes are critical to a service provider's ability to conduct business. Without accurate and up-to-date routes, the service provider might not be able to provide some of its service offerings.	• Internal management networks • Network operations prefixes • DNS resources
Network topology changes	These should be prioritized ahead of simple route refreshes.	• New router added to the network • Routers removed from the network

Properties of BGP Route Prioritization

BGP route prioritization in Junos OS is implemented using a set of 17 prioritized (numbered) output queues that are serviced by a user-configurable token mechanism. This section describes the prioritized output queues, the operation of the token system, and assignment of routes to queues.

Prioritized Output Queues

Table 2 on page 104 shows the available output queues and their function within the prioritization system. The prioritization system functions on a traditional low, medium, and high priority scale with 1 being the lowest priority and 16 being the highest priority.

Table 2: Prioritized Output Queues

Queue	Function
expedited	This is the highest priority output queue. Routes in this class are guaranteed some portion of the output queue processing while flushing the output queue. This queue has no number and is referred to in the configuration by its name.
1 (lowest priority)	This is the lowest priority output queue. This is the default priority queue, meaning that routes with no explicit queue assignment from either automatic protocol determination or user policy are placed in this queue by default. Route refresh messages are placed in this queue by default.
2 - 16 (low - high priority)	These output queues range in priority from lowest priority (2) to highest priority (16). They are assigned routes based on user policy or BGP peer configuration. Routes in a higher priority output queue can preempt the routes in lower priority queues.

Assignment of Routes to Queues

Assigning routes to the various queues can be accomplished by setting and assigning BGP export policies. This means that route priority can vary in each BGP peer group as well as in specific neighbor configurations within the BGP peer groups. You can also assign routes to queues using the action portion of a policy statement. Assignment of routes to queues by the action of a policy statement will override assignments made by BGP configuration.

When Routes are Assigned to Queues

Prioritized routes are added to their respective queues only during route processing. This generally occurs under the following conditions:

- When routes are received from established BGP neighbors.
- During a routing table walk, such as during reconfiguration or the initial establishment of a peer.

Routing table walks process routes sequentially in a table, based on the system's internal routing table order. For example, `inet.0`, `inet6.0`, and so on. This internal order is predetermined and cannot be modified by the operator.

If higher-priority routes reside in routing tables processed later in the order, these routes may be added to the queues after lower-priority routes encountered earlier in the process.

Work Token Mechanism

Tokens correspond to the work to create a BGP update message. All the queues are assigned tokens that are stored in buckets. The number of tokens in a given bucket is user-configurable. In this way, users can craft policies that permit their routes to be served in the proportions they prefer. The configuration of the priority scheduler is accomplished globally within BGP at the `[edit protocols bgp]` hierarchy level. By default, all priority queues have at least 1 token in their bucket to ensure that misconfigured priorities do not starve.

Understanding Queue Priority and Fairness

The scheme used by BGP route prioritization focuses on two elements: fairness and priority:

- Fairness means that when there is work to do in any given queue, other queues are guaranteed to get some work done at some point. How much work each queue is permitted to get done is determined by the number of tokens assigned to each priority.
- Priority means that when there is competing work and fairness has been ensured, to always choose the more important work.

For example, presume three classes of priority: low, medium, and high. These could be assigned to queues 1, 2, and 3, respectively. Alternatively, they could be assigned to queues 3, 6, and 9. For fairness, if the decision is that high priority gets 50% of the available work, medium gets 35%, and low gets the remaining 15%, tokens can be assigned as 50 to high, 35 to medium, and 15 to low. Alternatively, tokens can be assigned as 5 to high, 4 to medium, and 2 to low. You can assign any of the 17 queues any value between 1 and 100. The ratio of the number of tokens in a single queue to the total number of tokens in all queues gives the percentage of work that will be done in each queue.

Priority is most important when work appears in a queue while tokens are in the process of being spent in another queue by the work scheduler. Table 3 on page 106 shows the starting point for an example of this.

Table 3: Queues and Tokens

Priority Queue (Queue Number)	Number of Tokens Assigned to Queue	Number of Tokens Left in Queue	Number of Entries in Queue
High (9)	50	50	0
Medium (6)	35	15	5000
Low (3)	15	15	10000

If we assume that the work scheduler is processing the medium queue (queue number 6) and has spent 20 tokens, then there are 15 tokens left to be spent on the remaining entries in the medium queue and 15 tokens left to be spent in the low priority queue. If 5 entries arrive in the expedited queue prior to the next run of the work scheduler, those 5 entries will be sent first because there are still 50 tokens left in the expedited queue.

Queue Servicing Procedure

The queue servicing procedure operates per-BGP peer group with each group maintaining its own token buckets.

- Token buckets for each priority start full either at the configured number of tokens or at the default of 1.
- Each time a route entry is pulled from a queue to start a BGP update, a token is subtracted from that queue.
- While the expedited queue has tokens, every other queue entry is drawn from the expedited queue, subject to the route packing rules.
- Entries are taken from the queue that has the highest priority. This means that if entries are added to a higher priority queue between runs of the queue servicing mechanism, and there are tokens available in that higher priority queue, the new entries in the higher priority queue are sent first, thus preempting entries in lower priority queues. If the higher priority queue has no work tokens available when the new entries arrive, the new entries are not sent until after the next token refresh.
- Tokens are refreshed after all priority queues have been serviced (there are no entries remaining in any queue) or when all tokens are exhausted.

SEE ALSO

Example: Controlling Routing Table Convergence Using BGP Route Prioritization | 115

Example: Configuring the BGP Output Priority Scheduler and Global Address Family Priority

IN THIS SECTION

- Requirements | 107
- Overview | 107
- Configuration | 108
- Configure Global Output Priorities for a Route Family | 110
- Configure a BGP Group Named test1 | 111
- Verification | 112

This example shows how to configure and test the system-wide BGP route priority scheduler.

Requirements

This example uses the following hardware and software components:

- An MX Series router (R1) running Junos OS Release 16.1 or later

Before you configure the BGP route prioritization scheduler, be sure that the BGP protocol is running on the router.

Overview

The BGP route priority scheduler is used to control the amount of work done within the 17 output queues of the route prioritization system. The system uses a set of 17 prioritized output queues, per routing instance to which work tokens are assigned. All 17 prioritized output queues (1-16 and expedited) have 1 token assigned by default. Any number of tokens between 1 and 100 can be assigned to each of the 17 queues. Assigning tokens to the queues allows you to balance the amount of work performed on the routes within the queues. In addition, default settings for high, medium, and low priority queuing can be configured by assigning each keyword to a specific numbered output queue. In this example, we will configure each of the 17 priority queues with distinct numbers of work tokens and

we also configure global output priorities for inet unicast routes and demonstrate inheritance by setting up some BGP groups to override global priority settings.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **108**
- Configuring the Individual Output Priority Queues | **109**
- Configure Default Queues to Use for High, Medium, and Low Priority Route Updates | **110**
- Results | **110**

- Assign `update-tokens` to each of the 17 output queues.
- Specify which numbered queues will be used as the default `high`, `medium`, and `low` priority queues.
- Configure global output priorities for `inet unicast` routes.
- Configure a BGP group named test1 that will show group override capabilities.
- Configure a BGP group named test2 that will show global inheritance.

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.

```
set protocols bgp output-queue-priority expedited update-tokens 100
set protocols bgp output-queue-priority priority 1 update-tokens 1
set protocols bgp output-queue-priority priority 2 update-tokens 10
set protocols bgp output-queue-priority priority 3 update-tokens 15
set protocols bgp output-queue-priority priority 4 update-tokens 20
set protocols bgp output-queue-priority priority 5 update-tokens 25
set protocols bgp output-queue-priority priority 6 update-tokens 30
set protocols bgp output-queue-priority priority 7 update-tokens 35
set protocols bgp output-queue-priority priority 8 update-tokens 40
set protocols bgp output-queue-priority priority 9 update-tokens 45
set protocols bgp output-queue-priority priority 10 update-tokens 50
```

```
set protocols bgp output-queue-priority priority 11 update-tokens 55
set protocols bgp output-queue-priority priority 12 update-tokens 60
set protocols bgp output-queue-priority priority 13 update-tokens 65
set protocols bgp output-queue-priority priority 14 update-tokens 70
set protocols bgp output-queue-priority priority 15 update-tokens 75
set protocols bgp output-queue-priority priority 16 update-tokens 80
set protocols bgp output-queue-priority defaults low priority 1
set protocols bgp output-queue-priority defaults medium priority 10
set protocols bgp output-queue-priority defaults high expedited
set protocols bgp group reflector local-address 198.51.100.140
set protocols bgp family inet unicast output-queue-priority priority 1
set protocols bgp family inet unicast route-refresh-priority priority 2
set protocols bgp family inet unicast withdraw-priority priority 3
set protocols bgp group test1 family inet unicast output-queue-priority priority 4
set protocols bgp group test1 family inet unicast route-refresh-priority priority 6
set protocols bgp group test1 peer-as 64511 set protocols bgp group test1 local-as 64511
set protocols bgp group test1 neighbor 224.223.1.1
set protocols bgp group test1 neighbor 224.223.2.2 family inet unicast output-queue-priority
priority 7
set protocols bgp group test1 neighbor 224.223.2.2 family inet unicast route-refresh-priority
priority 8
set protocols bgp group test1 neighbor 224.223.2.2 family inet unicast withdraw-priority
expedited
set protocols bgp group test2 peer-as 64513
set protocols bgp group test2 local-as 64511
set protocols bgp group test2 neighbor 224.223.3.3
```

Configuring the Individual Output Priority Queues

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 Junos OS CLI User Guide.

1. Assign update tokens to each of the 17 prioritized output queues

```
[edit protocols bgp output-queue-priority]
user@R1# set expedited update-tokens 100
user@R1# set priority 1 update-tokens 1
user@R1# set priority 2 update-tokens 10
```

```
user@R1# set priority 3 update-tokens 15
user@R1# set priority 4 update-tokens 20
user@R1# set priority 5 update-tokens 25
user@R1# set priority 6 update-tokens 30
user@R1# set priority 7 update-tokens 35
user@R1# set priority 8 update-tokens 40
user@R1# set priority 9 update-tokens 45
user@R1# set priority 10 update-tokens 50
user@R1# set priority 11 update-tokens 55
user@R1# set priority 12 update-tokens 60
user@R1# set priority 13 update-tokens 65
user@R1# set priority 14 update-tokens 70
user@R1# set priority 15 update-tokens 75
user@R1# set priority 16 update-tokens 80
```

Configure Default Queues to Use for High, Medium, and Low Priority Route Updates

Step-by-Step Procedure

1.
```
[edit protocols bgp output-queue-priority]
user@R1# set defaults low priority 1
user@R1# set defaults medium priority 10
user@R1# set defaults high expedited
```

Results

To confirm the configuration, issue the `show bgp output-scheduler` command from operational mode:

Configure Global Output Priorities for a Route Family

IN THIS SECTION

- Procedure | 111

Procedure

Step-by-Step Procedure

1. Configure the global `output-queue-priority` for `inet unicast` routes:

```
[edit bgp family inet unicast]
user@R1# set output-queue-priority priority 1
user@R1# set route-refresh-priority priority 2
user@R1# set withdraw-priority priority 3
```

Configure a BGP Group Named test1

IN THIS SECTION

- Procedure | 111
- Configure a BGP Group Named test2 | 112

Procedure

Step-by-Step Procedure

1. Configure the group `test1` to override global output priorities and include one neighbor that overrides the group and one neighbor that does not.

```
[edit protocols bgp group test1]
user@R1# set family inet unicast output-queue-priority priority 4
user@R1# set family inet unicast route-refresh-priority priority 6
user@R1# set peer-as 64511
user@R1# set local-as 64511
user@R1# set neighbor 224.223.1.1
user@R1# set neighbor 224.223.2.2 family inet unicast output-queue-priority priority 7
user@R1# set neighbor 224.223.2.2 family inet unicast route-refresh-priority priority 8
user@R1# set neighbor 224.223.2.2 family inet unicast withdraw-priority expedited
```

Configure a BGP Group Named test2

Step-by-Step Procedure

1. Configure the BGP group `test2` to accept global defaults.

```
[edit protocols bgp group test2]
user@R1# set peer-as 64513
user@R1# set local-as 64511
user@R1# set neighbor 224.223.3.3
```

Verification

IN THIS SECTION

- Verifying the BGP Output Scheduler Configuration | **112**
- Verify Group Configuration, Group Override, and Neighbor Override | **113**
- Verify Inheritance from Global Priority Settings | **114**

Verifying the BGP Output Scheduler Configuration

Purpose

To verify the configuration of the BGP output scheduler, issue the `show bgp output-scheduler` command from operational mode.

Action

```
user@R1> show bgp output-scheduler
user@R1> show bgp output-scheduler
Instance: master
  Class         Tokens
  ------------  ------
  Priority  1        1
  Priority  2       10
  Priority  3       15
```

```
  Priority  4       20
  Priority  5       25
  Priority  6       30
  Priority  7       35
  Priority  8       40
  Priority  9       45
  Priority 10       50
  Priority 11       55
  Priority 12       60
  Priority 13       65
  Priority 14       70
  Priority 15       75
  Priority 16       80
  Expedited        100

  Priority  Class
  --------  ------------
  low       Priority  1
  medium    Priority 10
  high      Expedited
```

Meaning

The output shows that the output scheduler configuration was successful in applying the proper number of tokens to each output queue and that the high, medium, and low priority keywords were assigned to the proper output queues.

Verify Group Configuration, Group Override, and Neighbor Override

Purpose

To verify that the configured groups demonstrate group override, neighbor override and inheritance, issue the `show bgp group` *`group-name`* command from operational mode.

Action

```
user@R1> show bgp group test1
Group Type: Internal    AS: 64511                Local AS: 64511
  Name: test1           Index: 2                 Flags: <>
  Options: <LocalAS>
  Holdtime: 0
```

```
  NLRI inet-unicast:
    OutQ: priority 7 RRQ: priority 8 WDQ: expedited
  Local AS: 64511 Local System AS: 64511
  Total peers: 1        Established: 0
  224.223.2.2

Group Type: Internal    AS: 64511                 Local AS: 64511
  Name: test1           Index: 1                  Flags: <Export Eval>
  Options: <LocalAS>
  Holdtime: 0
  NLRI inet-unicast:
    OutQ: priority 4 RRQ: priority 6 WDQ: priority 3
  Local AS: 64511 Local System AS: 64511
  Total peers: 1        Established: 0
  224.223.1.1
```

Meaning

The output shows that the output queue priority for peer 224.223.2.2 is 7, the route refresh priority is 8, and the withdraw priority is expedited. While the output queue priority for neighbor 224.223.1.1 is 4, the route refresh priority is 6, and the withdraw priority is the default setting for the family `inet unicast`, or 3.

Verify Inheritance from Global Priority Settings

Purpose

To verify that groups that are not configured to override the global BGP route prioritization settings, issue the `show bgp group` *`group-name`* command at the operational level.

Action

```
user@R1> show bgp group test2
Group Type: External                            Local AS: 64511
  Name: test2           Index: 3                Flags: <Export Eval>
  Options: <LocalAS>
  Holdtime: 0
  NLRI inet-unicast:
    OutQ: priority 1 RRQ: priority 2 WDQ: priority 3
  Local AS: 64511 Local System AS: 64511
```

```
Total peers: 1        Established: 0
224.223.3.3
```

Meaning

The output shows that the default route priorities for `inet unicast` routes in the `test2` group match the global configuration.

SEE ALSO

Understanding BGP Route Prioritization | 101

Example: Controlling Routing Table Convergence Using BGP Route Prioritization

IN THIS SECTION

- Requirements | 115
- Overview | 116
- Configure BGP Route Prioritization | 116
- Verification | 119

The following example configures BGP route prioritization in order to allow `inet labeled-unicast` routes to converge before `inet unicast` routes.

Requirements

This example uses the following hardware and software components:

- An MX-Series router (R1) running Junos OS Release 16.1 or later that will be the focus of the example.
- A second router (R2) configured as an internal BGP peer with R1.
- A BGP route reflector (RR) that will be used to populate the routing tables of R1. In this example, we will not configure the route reflector.

Overview

The BGP route prioritization feature is designed to allow the prioritization of outbound BGP update messages in a router. Using BGP route prioritization enables the user to ensure that more important BGP route updates, such as GRE or MPLS tunnel endpoint changes, are sent out before less important BGP route updates, such as route refresh updates.

In this example, we will configure R1 to treat `inet labeled-unicast` route updates to R2 as higher priority than `inet unicast` route updates. To do this, we will configure the R2 router to accept both `inet unicast` and `inet labeled-unicast` routes from its peer router, R1. Then we will populate the `inet.0` routing table on R1 from a route reflector and import a portion of that table into the `labeled-unicast` table, `inet.3` using `rib-group` import. As the routes are queued on R1, we can validate the operation by observing whether the routes in the `inet.3` RIB are flushed before the remainder of the routes in the `inet.0` RIB.

Configure BGP Route Prioritization

IN THIS SECTION

- CLI Quick Configuration | **116**
- Procedure | **118**

Configure R2 as a BGP peer of R1.

On R1:

- Configure the router R2 as a peer of router R1.
- Create a BGP group named reflector that will be used to obtain Internet routes from a route reflector.
- Create a BGP group named internal that will be used for assigning the labeled-unicast traffic to a higher priority output-queue.
- Create a RIB group into which the routes received from the reflector are imported.
- Create the policy that determines what portion of the `inet.0` RIB is imported into the RIB group.

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 R2

```
set protocols bgp group internal type internal
set protocols bgp group internal family inet unicast
set protocols bgp group internal family inet labeled-unicast rib inet.3
set protocols bgp group internal peer-as 64511
set protocols bgp group internal local-as 64511
set protocols bgp group internal  neighbor 192.0.2.1
```

Router R1

```
set protocols bgp group internal type internal
set protocols bgp group internal hold-time 900
set protocols bgp group internal family inet unicast withdraw-priority expedited
set protocols bgp group internal family inet labeled-unicast output-queue-priority priority 2
set protocols bgp group internal family inet labeled-unicast rib inet.3
set protocols bgp group internal family inet-vpn unicast
set protocols bgp group internal local-as 64511
set protocols bgp group internal neighbor 192.0.2.2 local-address 192.0.2.1
set protocols bgp group reflector local-address 203.0.113.225
set protocols bgp group reflector family inet unicast rib-group into3
set protocols bgp group reflector peer-as 64500
set protocols bgp group reflector local-as 64496
set protocols bgp group reflector neighbor 192.51.100.71 multihop
set policy-options policy-statement match-all then accept
set routing-options rib-groups into3 import-rib inet.0
set routing-options rib-groups into3 import-rib inet.3
set routing-options rib-groups into3 import-policy match-long
set policy-options policy-statement match-long term a from route-filter 192.0.0.0/8 prefix-
length-range /20-/24
set policy-options policy-statement match-long term a then accept
set policy-options policy-statement match-long then reject
set policy-options policy-statement match-all then accept
```

Procedure

Step-by-Step Procedure

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

To configure R2:

1. Configure a BGP group named internal.

```
[edit protocols bgp group internal]
user@R2# set type internal
user@R2# set  family inet unicast
user@R2# set  family inet labeled-unicast rib inet.3
user@R2# set  peer-as 64511
user@R2# set  local-as 64511
user@R2# set  neighbor 192.0.2.1
```

Step-by-Step Procedure

To configure R1:

1. Configure a BGP group named reflector that receives routes from the RR.

```
[edit protocols bgp group reflector]
user@R1# set local-address 203.0.113.225
user@R1# set family inet unicast rib-group into3
user@R1# set peer-as 64500
user@R1# set local-as 64496
user@R1# set neighbor 192.51.100.71 multihop
```

2. Configure a BGP group named internal

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set hold-time 900
user@R1# set family inet unicast withdraw-priority expedited
user@R1# set family inet labeled-unicast output-queue-priority priority 2
user@R1# set family inet labeled-unicast rib inet.3
```

```
user@R1# set family inet-vpn unicast
user@R1# set local-as 64511
user@R1# set neighbor 192.0.2.2 local-address 192.0.2.1
```

3. Configure a RIB group named into3

```
[edit routing-options rib-groups into3]
user@R1# set import-rib inet.0
user@R1# set import-rib inet.3
user@R1# set import-policy match-long
```

4. Configure a routing policy named match-long

```
[edit policy-options policy-statement match-long]
user@R1# set term a from route-filter 192.0.0.0/8 prefix-length-range /20-/24
user@R1# set term a then accept
user@R1# set then reject
```

5. Configure a routing policy named match-all

```
[edit policy-options policy-statement match-all]
user@R1# set then accept
```

Verification

IN THIS SECTION

- Verifying that Neighbor Updates are Properly Prioritized | 119

Verifying that Neighbor Updates are Properly Prioritized

Purpose

To confirm that route updates are being placed in the proper queues and that the queues are updating.

Action

To see the route updates that are queued for the BGP neighbor 192.0.2.2, issue the `show bgp neighbor output-queue 192.0.2.2` command from operational mode. The outputs show that inet-unicast routes are advertised before inet-labeled-unicast routes.

```
user@R1> show bgp neighbor output-queue 192.0.2.2
Peer: 192.0.2.2+179 AS 64511 Local: 192.0.2.1+63704 AS 64511
Output Queue[0]: 502701 (inet.0, inet-unicast)
Priority 1 : 502701
Priority 2 : 0
Priority 3 : 0
Priority 4 : 0
Priority 5 : 0
Priority 6 : 0
Priority 7 : 0
Priority 8 : 0
Priority 9 : 0
Priority 10: 0
Priority 11: 0
Priority 12: 0
Priority 13: 0
Priority 14: 0
Priority 15: 0
Priority 16: 0
Expedited : 0
```

```
user@R1> show bgp neighbor output-queue 192.0.2.2
Peer: 192.0.2.2+179 AS 64511 Local: 192.0.2.1+63704 AS 64511
Output Queue[1]: 6687 (inet.3, inet-labeled-unicast)
Priority 1 : 0
Priority 2 : 6687
Priority 3 : 0
Priority 4 : 0
Priority 5 : 0
Priority 6 : 0
Priority 7 : 0
Priority 8 : 0
Priority 9 : 0
Priority 10: 0
Priority 11: 0
```

```
Priority 12: 0
Priority 13: 0
Priority 14: 0
Priority 15: 0
Priority 16: 0
Expedited : 0
user@R1> show bgp neighbor output-queue 192.0.2.2
Peer: 192.0.2.2+179 AS 64511 Local: 192.0.2.1+63704 AS 64511
Output Queue[0]: 491187 (inet.0, inet-unicast)
Priority 1 : 491187
Priority 2 : 0
Priority 3 : 0
Priority 4 : 0
Priority 5 : 0
Priority 6 : 0
Priority 7 : 0
Priority 8 : 0
Priority 9 : 0
Priority 10: 0
Priority 11: 0
Priority 12: 0
Priority 13: 0
Priority 14: 0
Priority 15: 0
Priority 16: 0
Expedited : 0
user@R1> show bgp neighbor output-queue 192.0.2.2
Peer: 192.0.2.2+179 AS 64511 Local: 192.0.2.1+63704 AS 64511
Output Queue[1]: 0 (inet.3, inet-labeled-unicast)
Priority 1 : 0
Priority 2 : 0
Priority 3 : 0
Priority 4 : 0
Priority 5 : 0
Priority 6 : 0
Priority 7 : 0
Priority 8 : 0
Priority 9 : 0
Priority 10: 0
Priority 11: 0
Priority 12: 0
Priority 13: 0
Priority 14: 0
```

```
Priority 15: 0
Priority 16: 0
Expedited : 0
```

Meaning

The output from `show bgp neighbor output-queue 192.0.2.2` shows that the labeled unicast route updates are placed in the priority 2 output queue and that the priority 2 output queue is emptied before the unicast route updates that are in the priority 1 output queue.

SEE ALSO

Example: Configuring the BGP Output Priority Scheduler and Global Address Family Priority | **107**

Understanding BGP Route Prioritization | **101**

BGP Auto-Discovered Neighbors

IN THIS SECTION

- Understanding BGP Auto-discovered Neighbor | **123**
- Example: Configuring BGP Auto-discovered Neighbor | **124**

Understanding BGP Auto-discovered Neighbor

SUMMARY

Use BGP auto-discovered neighbor to configure BGP peering by interface rather than by specifying remote or local neighbor IP addresses.

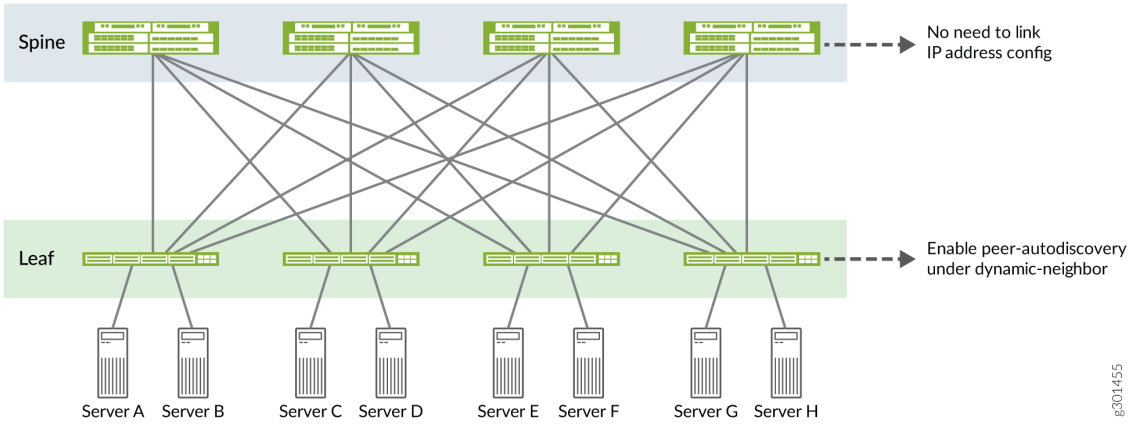
IN THIS SECTION

- Benefits of BGP Auto-discovered Neighbor | **124**
- Limitations | **124**

To establish a BGP session between routers, you must explicitly configure BGP groups and peers by address. BGP peering sessions require that you identify source and destination IP addresses for endpoints of the TCP communication. Therefore, explicitly configuring these addresses is an obstacle to network scale-out and an opportunity for misconfiguration.

To streamline your BGP configuration, we have removed the need to configure per-peer address from BGP. Use BGP auto-discovered neighbor to configure BGP peering by interface rather than by specifying remote or local neighbor IP addresses. This includes use of implicit or protocol mechanisms to discover the IP addresses for use in the TCP peering sessions.

Figure 8: BGP Auto-discovered Neighbor



NOTE: Peering behavior and address usage must be explicitly configured to avoid peering changes based on interface address changes due to configuration or address validity (for example, IPv6 Duplicate Address Detection (DAD)).

BGP determines the address families to peer over based on the configuration. The peering sessions come up based on availability of the interface addresses for the determined families. The peer link-local address is discovered using IPv6 neighbor discovery (RFC4861) and creates a BGP session toward that neighbor. A link-local address is generated even when IPv6 interfaces have no addresses configured.

NOTE: You must enable IPv6 neighbor discovery for this feature to work.

Benefits of BGP Auto-discovered Neighbor

- Simplifies IGP deployment to a single-hop external BGP (EBGP)
- Configures neighbors by interfaces and interface ranges instead of by IP addresses
- Minimizes configuration on both sides with dynamic-neighbor groups

Limitations

The feature has the following limitations:

- Scoped BGP link-local addresses do not currently support allow neighbor ranges.
- BGP auto-discovered neighbor feature does not support IBGP or multi-hop EBGP.
- BGP auto-discovered neighbor does not support discovering or peering with more than one remote neighbor on any given interface.

Example: Configuring BGP Auto-discovered Neighbor

SUMMARY

This example shows how to configure BGP Auto-discovered Neighbor.

IN THIS SECTION

Overview | 125

- Requirements | **126**
- Configuration | **126**
- Verification | **135**

Overview

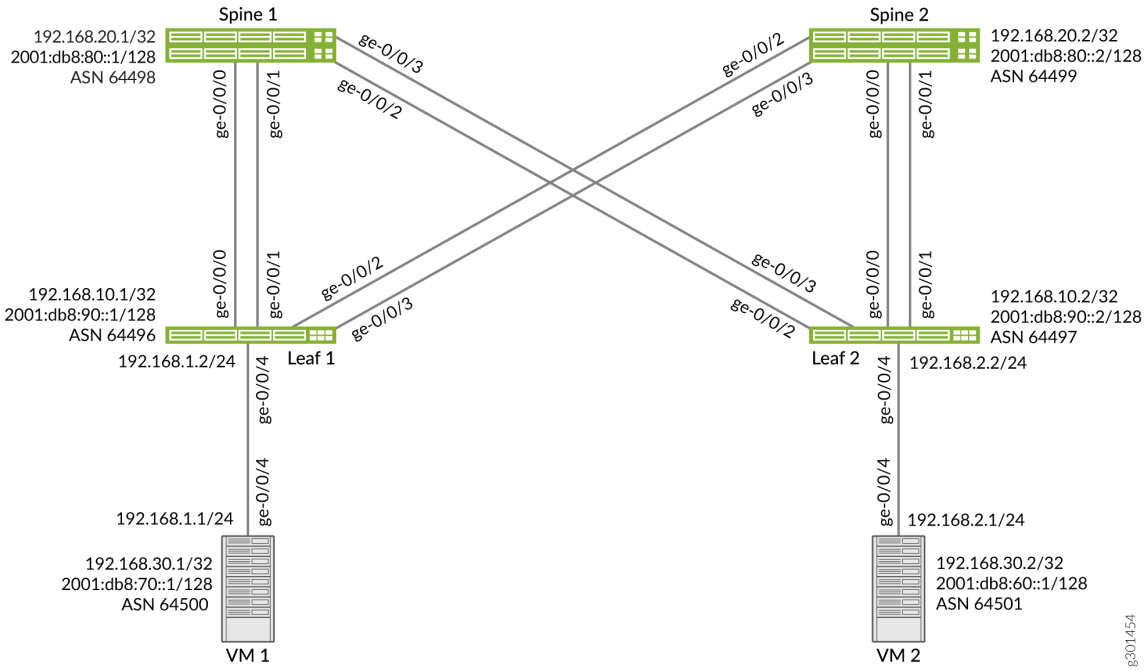
IN THIS SECTION

- Topology | **125**

Starting in Junos OS Release 21.1R1, we support BGP auto-discovered neighbors using IPv6 Neighbor Discovery Protocol (ND). This feature allows BGP to create peer neighbor sessions using link-local IPv6 addresses of directly connected neighbor routers. You need not specify remote or local neighbor IP addresses.

Topology

The following figure shows a simplified sample topology.



Requirements

This example uses the following hardware and software components:

MX Series routers

Junos OS Release 21.1R1 or later

Configuration

IN THIS SECTION

- CLI Quick Configuration | **127**
- Configuring VM1 | **131**
- Results | **132**

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.

VM1

```
set interfaces interface-range tor-to-leaf member ge-0/0/4
set interfaces interface-range tor-to-leaf unit 0 family inet6
set interfaces ge-0/0/4 unit 0 family inet address 192.168.1.1/24
set interfaces lo0 unit 0 family inet address 192.168.30.1/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:70::1/128
set policy-options policy-statement DIRECT-RTS from protocol direct
set policy-options policy-statement DIRECT-RTS then accept
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64500
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface tor-to-leaf
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface tor-to-leaf
set protocols bgp group autodisc peer-as-list a-list
set protocols bgp group to-leaf-v4 family inet unicast extended-nexthop
set protocols bgp group to-leaf-v4 export DIRECT-RTS
set protocols bgp group to-leaf-v4 local-as 64500
set protocols bgp group to-leaf-v4 neighbor 192.168.1.2 peer-as 64496
```

VM2

```
set interfaces interface-range tor-to-leaf member ge-0/0/4
set interfaces interface-range tor-to-leaf unit 0 family inet6
set interfaces ge-0/0/4 unit 0 family inet address 192.168.2.1/24
set interfaces lo0 unit 0 family inet address 192.168.30.2/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:60::1/128
set policy-options policy-statement DIRECT-RTS from protocol direct
set policy-options policy-statement DIRECT-RTS then accept
```

```
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64501
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface tor-to-leaf
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface tor-to-leaf
set protocols bgp group autodisc peer-as-list a-list
set protocols bgp group to-leaf-v4 family inet unicast extended-nexthop
set protocols bgp group to-leaf-v4 export DIRECT-RTS
set protocols bgp group to-leaf-v4 local-as 64501
set protocols bgp group to-leaf-v4 neighbor 192.168.2.2 peer-as 64497
```

Leaf 1

```
set interfaces interface-range to-spine member "ge-0/0/[0-4]"
set interfaces interface-range to-spine unit 0 family inet
set interfaces interface-range to-spine unit 0 family inet6
set interfaces ge-0/0/4 unit 0 family inet address 192.168.1.2/24
set interfaces lo0 unit 0 family inet address 192.168.10.1/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:90::1/128
set policy-options policy-statement DIRECT-RTS from protocol direct
set policy-options policy-statement DIRECT-RTS then accept
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64496
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface to-spine max-advertisement-interval 9
set protocols router-advertisement interface to-spine min-advertisement-interval 3
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface to-spine
```

```
set protocols bgp group autodisc peer-as-list a-list
set protocols bgp group to-crpd-v4 family inet unicast extended-nexthop
set protocols bgp group to-crpd-v4 export DIRECT-RTS
set protocols bgp group to-crpd-v4 neighbor 192.168.1.1 peer-as 64500
```

Leaf 2

```
set interfaces interface-range to-spine member "ge-0/0/[0-4]"
set interfaces interface-range to-spine unit 0 family inet
set interfaces interface-range to-spine unit 0 family inet6
set interfaces ge-0/0/4 unit 0 family inet address 192.168.2.2/24
set interfaces lo0 unit 0 family inet address 192.168.10.2/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:90::2/128
set policy-options policy-statement DIRECT-RTS from protocol direct
set policy-options policy-statement DIRECT-RTS then accept
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface to-spine max-advertisement-interval 9
set protocols router-advertisement interface to-spine min-advertisement-interval 3
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface to-spine
set protocols bgp group autodisc peer-as-list a-list
set protocols bgp group to-crpd-v4 family inet unicast extended-nexthop
set protocols bgp group to-crpd-v4 export DIRECT-RTS
set protocols bgp group to-crpd-v4 neighbor 192.168.2.1 peer-as 64501
```

Spine 1

```
set interfaces interface-range to-leaf member "ge-0/0/[0-3]"
set interfaces interface-range to-leaf unit 0 family inet
set interfaces interface-range to-leaf unit 0 family inet6
set interfaces lo0 unit 0 family inet address 192.168.20.1/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:80::1/128
set policy-options policy-statement DIRECT-RTS from protocol direct
```

```
set policy-options policy-statement DIRECT-RTS then accept
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64498
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface to-leaf max-advertisement-interval 9
set protocols router-advertisement interface to-leaf min-advertisement-interval 3
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface to-leaf
set protocols bgp group autodisc peer-as-list a-list
```

Spine 2

```
set interfaces interface-range to-leaf member "ge-0/0/[0-3]"
set interfaces interface-range to-leaf unit 0 family inet
set interfaces interface-range to-leaf unit 0 family inet6
set interfaces lo0 unit 0 family inet address 192.168.20.2/32
set interfaces lo0 unit 0 family inet6 address 2001:db8:80::2/128
set policy-options policy-statement DIRECT-RTS from protocol direct
set policy-options policy-statement DIRECT-RTS then accept
set policy-options policy-statement lb then load-balance per-packet
set policy-options as-list a-list members 1-65535
set routing-options autonomous-system 64499
set routing-options forwarding-table export lb
set routing-options forwarding-table ecmp-fast-reroute
set protocols router-advertisement interface to-leaf max-advertisement-interval 9
set protocols router-advertisement interface to-leaf min-advertisement-interval 3
set protocols bgp group autodisc family inet unicast extended-nexthop
set protocols bgp group autodisc family inet6 unicast
set protocols bgp group autodisc export DIRECT-RTS
set protocols bgp group autodisc multipath multiple-as
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family inet6 ipv6-nd
set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface to-leaf
set protocols bgp group autodisc peer-as-list a-list
```

Configuring VM1

To configure BGP auto-discovered neighbor, perform the following steps on VM1:

1. Configure the device interfaces.

```
user@VM1# set interfaces interface-range tor-to-leaf member ge-0/0/4
user@VM1# set interfaces interface-range tor-to-leaf unit 0 family inet6
```

2. Create the loopback interface and configure the IP address.

```
user@VM1# set interfaces lo0 unit 0 family inet address 192.168.30.1/32
user@VM1# set interfaces lo0 unit 0 family inet6 address 2001:db8:70::1/128
```

3. Enable routing policies.

```
user@VM1# set policy-options policy-statement DIRECT-RTS from protocol direct
user@VM1# set policy-options policy-statement DIRECT-RTS then accept
user@VM1# set policy-options policy-statement lb then load-balance per-packet
user@VM1# set policy-options as-list a-list members 1-65535
```

4. Configure the autonomous system (AS) number

```
user@VM1# set routing-options autonomous-system 64500
```

5. Apply the per-packet policy to enable load balancing of traffic and ECMP.

```
user@VM1# set routing-options forwarding-table export lb
user@VM1# set routing-options forwarding-table ecmp-fast-reroute
```

6. Configure BGP to establish internal and external peering sessions.

```
user@VM1# set protocols bgp group autodisc family inet unicast extended-nexthop
user@VM1# set protocols bgp group autodisc family inet6 unicast
user@VM1# set protocols bgp group autodisc export DIRECT-RTS
user@VM1# set protocols bgp group autodisc multipath multiple-as
user@VM1# set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery family
inet6 ipv6-nd
```

```
user@VM1# set protocols bgp group autodisc dynamic-neighbor ndp peer-auto-discovery interface
tor-to-leaf
user@VM1# set protocols bgp group autodisc peer-as-list a-list
user@VM1# set protocols bgp group to-leaf-v4 family inet unicast extended-nexthop
user@VM1# set protocols bgp group to-leaf-v4 export DIRECT-RTS
user@VM1# set protocols bgp group to-leaf-v4 local-as 5
user@VM1# set protocols bgp group to-leaf-v4 neighbor 192.168.1.2 peer-as 1
```

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

```
user@VM1# commit
```

Results

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

```
[edit]
user@VM1# show interfaces
interface-range tor-to-leaf {
    member ge-0/0/4;
    unit 0 {
        family inet6;
    }
}
ge-0/0/4 {
    unit 0 {
        family inet {
            address 192.168.1.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.30.1/32;
        }
        family inet6 {
            address 2001:db8:70::1/128;
```

```
        }
    }
}
```

```
[edit]
user@VM1# show protocols
router-advertisement {
    interface tor-to-leaf;
}
bgp {
    group autodisc {
        family inet {
            unicast {
                extended-nexthop;
            }
        }
        family inet6 {
            unicast;
        }
        export DIRECT-RTS;
        multipath {
            multiple-as;
        }
        dynamic-neighbor ndp {
            peer-auto-discovery {
                family inet6 {
                    ipv6-nd;
                }
                interface tor-to-leaf;
            }
        }
        peer-as-list a-list;
    }
    group to-leaf-v4 {
        family inet {
            unicast {
                extended-nexthop;
            }
        }
        export DIRECT-RTS;
        local-as 64500;
```

```
        neighbor 192.168.1.2 {
            peer-as 64496;
        }
    }
}
```

```
[edit]
user@VM1# show policy-options
policy-statement DIRECT-RTS {
    from protocol direct;
    then accept;
}
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
as-list a-list members 1-65535;
```

```
[edit]
user@VM1# show policy-options
policy-statement DIRECT-RTS {
    from protocol direct;
    then accept;
}
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
as-list a-list members 1-65535;

[
```

```
edit]
user@VM1# show routing-options
autonomous-system 64500;
forwarding-table {
    export lb;
```

```
    ecmp-fast-reroute;
}
```

Verification

IN THIS SECTION

- Verifying Auto-discovered neighbors | **135**
- Verifying BGP Auto-discovered Peers | **137**

Confirm that the configuration is working properly.

Verifying Auto-discovered neighbors

IN THIS SECTION

- Purpose | **135**
- Action | **135**
- Meaning | **137**

Purpose

Verify the auto-discovered BGP neighbors.

Action

From operational mode, run the `show bgp summary auto-discovered` command

On Leaf1

```
user@Leaf1> show bgp summary auto-discovered
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 2 Peers: 5 Down peers: 1
Auto-discovered peers: 4
```

```
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                     24         20          0          0          0          0
inet6.0
                     16         16          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
fe80::5668:a3ff:fe16:1049%ge-0/0/3.0       64499        194        195       0       1
1:25:18 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
fe80::5668:a3ff:fe16:104c%ge-0/0/4.0       64499        193        195       0       1
1:25:18 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
fe80::5668:a3ff:fe16:12c9%ge-0/0/1.0       64498        217        223       0       1
1:35:53 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
fe80::5668:a3ff:fe16:12ce%ge-0/0/2.0       64498        218        223       0       1
1:35:57 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
```

On Spine1

```
user@Spine1> show bgp summary auto-discovered
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 4 Down peers: 0
Auto-discovered peers: 4
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                     24         20          0          0          0          0
inet6.0
                     16         16          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
fe80::5668:a3ff:fe16:2e7%ge-0/0/0.0       64496        245        237       0       0
1:45:19 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
```

```
fe80::5668:a3ff:fe16:2f2%ge-0/0/1.0       64496        245        238       0       0
1:45:23 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
fe80::5668:a3ff:fe16:e49%ge-0/0/2.0       64497        245        243       0       0
1:45:19 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
fe80::5668:a3ff:fe16:e59%ge-0/0/3.0       64497        247        248       0       0
1:45:23 Establ
  inet.0: 5/6/6/0
  inet6.0: 4/4/4/0
```

Meaning

The output shows the summary of auto-discovered bgp neighbors. You can see the number of auto-discovered peers and its details.

Verifying BGP Auto-discovered Peers

IN THIS SECTION

- Purpose | **137**
- Action | **137**
- Meaning | **139**

Purpose

Verify the auto-discovered BGP neighbors.

Action

From operational mode, run the `show bgp neighbor auto-discovered` command.

On Leaf1

```
user@Leaf1> show bgp neighbor auto-discovered
Peer: fe80::5668:a3ff:fe16:1049%ge-0/0/3.0+65265 AS 64499 Local:
```

```
fe80::5668:a3ff:fe16:2f6%ge-0/0/3.0+179 AS 64496
  Group: autodisc              Routing-Instance: master
  Forwarding routing-instance: master
  Type: External    State: Established    Flags: <Sync PeerAsList AutoDiscoveredNdp>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ DIRECT-RTS ]
  Options: <AddressFamily Multipath Refresh>
  Options: <MultipathAs>
  Options: <GracefulShutdownRcv>
  Address families configured: inet-unicast inet6-unicast
  Holdtime: 90 Preference: 170
  Graceful Shutdown Receiver local-preference: 0
  Number of flaps: 1
  Last flap event: RecvNotify
  Error: 'Cease' Sent: 0 Recv: 1
  Peer ID: 128.49.102.24   Local ID: 128.49.102.139    Active Holdtime: 90
  Keepalive Interval: 30         Group index: 2    Peer index: 2    SNMP index: 9
  I/O Session Thread: bgpio-0 State: Enabled
  BFD: disabled, down
  Local Interface: ge-0/0/3.0
  NLRI for restart configured on peer: inet-unicast inet6-unicast
  NLRI advertised by peer: inet-unicast inet6-unicast
  NLRI for this session: inet-unicast inet6-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  Restart flag received from the peer: Notification
  NLRI that restart is negotiated for: inet-unicast inet6-unicast
  NLRI of received end-of-rib markers: inet-unicast inet6-unicast
  NLRI of all end-of-rib markers sent: inet-unicast inet6-unicast
  ..................
```

On Spine1

```
user@Spine1>  show bgp neighbor auto-discovered
Peer: fe80::5668:a3ff:fe16:2e7%ge-0/0/0.0+60458 AS 64496 Local:
fe80::5668:a3ff:fe16:12c9%ge-0/0/0.0+179 AS 64498
  Group: autodisc              Routing-Instance: master
  Forwarding routing-instance: master
  Type: External    State: Established    Flags: <Sync PeerAsList AutoDiscoveredNdp>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
```

```
Last Error: None
Export: [ DIRECT-RTS ]
Options: <AddressFamily Multipath Refresh>
Options: <MultipathAs>
Options: <GracefulShutdownRcv>
Address families configured: inet-unicast inet6-unicast
Holdtime: 90 Preference: 170
Graceful Shutdown Receiver local-preference: 0
Number of flaps: 0
Peer ID: 128.49.102.139  Local ID: 128.49.103.129    Active Holdtime: 90
Keepalive Interval: 30         Group index: 1    Peer index: 3    SNMP index: 7
I/O Session Thread: bgpio-0 State: Enabled
BFD: disabled, down
Local Interface: ge-0/0/0.0
NLRI for restart configured on peer: inet-unicast inet6-unicast
NLRI advertised by peer: inet-unicast inet6-unicast
NLRI for this session: inet-unicast inet6-unicast
Peer supports Refresh capability (2)
Stale routes from peer are kept for: 300
Peer does not support Restarter functionality
Restart flag received from the peer: Notification
NLRI that restart is negotiated for: inet-unicast inet6-unicast
NLRI of received end-of-rib markers: inet-unicast inet6-unicast
NLRI of all end-of-rib markers sent: inet-unicast inet6-unicast
Peer does not support LLGR Restarter functionality
Peer supports 4 byte AS extension (peer-as 64496)
Peer does not support Addpath
NLRI that we support extended nexthop encoding for: inet-unicast
NLRI that peer supports extended nexthop encoding for: inet-unicast
Table inet.0 Bit: 20000
.....................
```

Meaning

The output shows information about the auto-discovered BGP neighbors.

3

CHAPTER

Configuring BGP Session Attributes

IN THIS CHAPTER

- Autonomous Systems for BGP Sessions | **141**
- Local Preference for BGP Routes | **276**
- BGP 4-Byte AS Numbers | **326**
- BGP MED Attribute | **353**
- BGP Multihop Sessions | **412**

Autonomous Systems for BGP Sessions

IN THIS SECTION

- Understanding the BGP Local AS Attribute | **141**
- Example: Configuring a Local AS for EBGP Sessions | **146**
- Example: Configuring a Private Local AS for EBGP Sessions | **162**
- Understanding the Accumulated IGP Attribute for BGP | **171**
- Example: Configuring the Accumulated IGP Attribute for BGP | **172**
- Understanding AS Override | **227**
- Example: Configuring a Layer 3 VPN with Route Reflection and AS Override | **228**
- Example: Enabling BGP Route Advertisements | **242**
- Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection | **254**
- Example: Ignoring the AS Path Attribute When Selecting the Best Path | **255**
- Understanding Private AS Number Removal from AS Paths | **265**
- Example: Removing Private AS Numbers from AS Paths | **267**

Understanding the BGP Local AS Attribute

When an Internet service provider (ISP) acquires a network that belongs to a different autonomous system (AS), there is no seamless method for moving the BGP peers of the acquired network to the AS of the acquiring ISP. The process of configuring the BGP peers with the new AS number can be time-consuming and cumbersome. Sometimes customers do not want to or are not immediately able to modify their peer arrangements or configuration. During this kind of transition period, it can be useful to configure BGP-enabled devices in the new AS to use the former AS number in BGP updates. This former AS number is called a *local AS*.

Using a local AS number permits the routing devices in an acquired network to appear to belong to the former AS.

For example, ISP A, with an AS of 65200, acquires ISP B, with an AS of 65250. ISP B has a customer, ISP C, that does not want to change its configuration. After ISP B becomes part of ISP A, a local AS number of 65250 is configured for use in EBGP peer sessions with ISP C. Consequently, the local AS number of 65250 is either prepended before or used instead of the global AS number of 65200 in the AS path used to export routes to direct external peers in ISP C.

If the route is received from an internal BGP (IBGP) peer, the AS path includes the local AS number prepended before the global AS number.

The local AS number is used instead of the global AS number if the route is an external route, such as a static route or an interior gateway protocol (IGP) route that is imported into BGP. If the route is external and you want the global AS number to be included in the AS path, you can apply a routing policy that uses `as-path-expand` or `as-path-prepend`. Use the `as-path-expand` policy action to place the global AS number behind the local AS number. Use the `as-path-prepend` policy action to place the global AS number in front of the local AS number.

For example:

```
user@R2# show policy-options
policy-statement prepend-global {
    term 1 {
        from protocol static;
        then {
            as-path-prepend 65200; # or use as-path-expand
            accept;
        }
    }
}
```

```
user@R2# show protocols bgp
group ext {
    export prepend-global;
    type external;
    local-as 65250;
    neighbor 10.0.0.1 {
        peer-as 65100;
    }
    neighbor 10.1.0.2 {
        peer-as 65300;
    }
}
```

```
user@R2# show routing-options
static {
    route 10.1.1.1/32 next-hop 10.0.0.1;
```

```
}
autonomous-system 65200;
```

```
user@R3# run show route 10.1.1.1 protocol bgp
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.1/32         *[BGP/170] 00:05:11, localpref 100
                      AS path: 65200 65250 I, validation-state: unverified
                    > to 10.1.0.1 via lt-1/2/0.4
```

In a Layer 3 VPN scenario, in which a provider edge (PE) device uses external BGP (EBGP) to peer with a customer edge (CE) device, the `local-as` statement behaves differently than in the non-VPN scenario. In the VPN scenario, the global AS number defined in the master instance is prepended to the AS path by default. To override this behavior, you can configure the `no-prepend-global-as` in the routing-instance BGP configuration on the PE device, as shown here:

```
user@R2# show routing-instances
red {
    instance-type vrf;
    interface fe-1/2/0.2;
    route-distinguisher 10:1;
    vrf-target target:10:1;
    protocols {
        bgp {
            group toR1 {
                type external;
                peer-as 65001;
                local-as 65200 no-prepend-global-as;
                neighbor 10.1.1.1;
            }
        }
    }
}
```

The Junos operating system (Junos OS) implementation of the local AS attribute supports the following options:

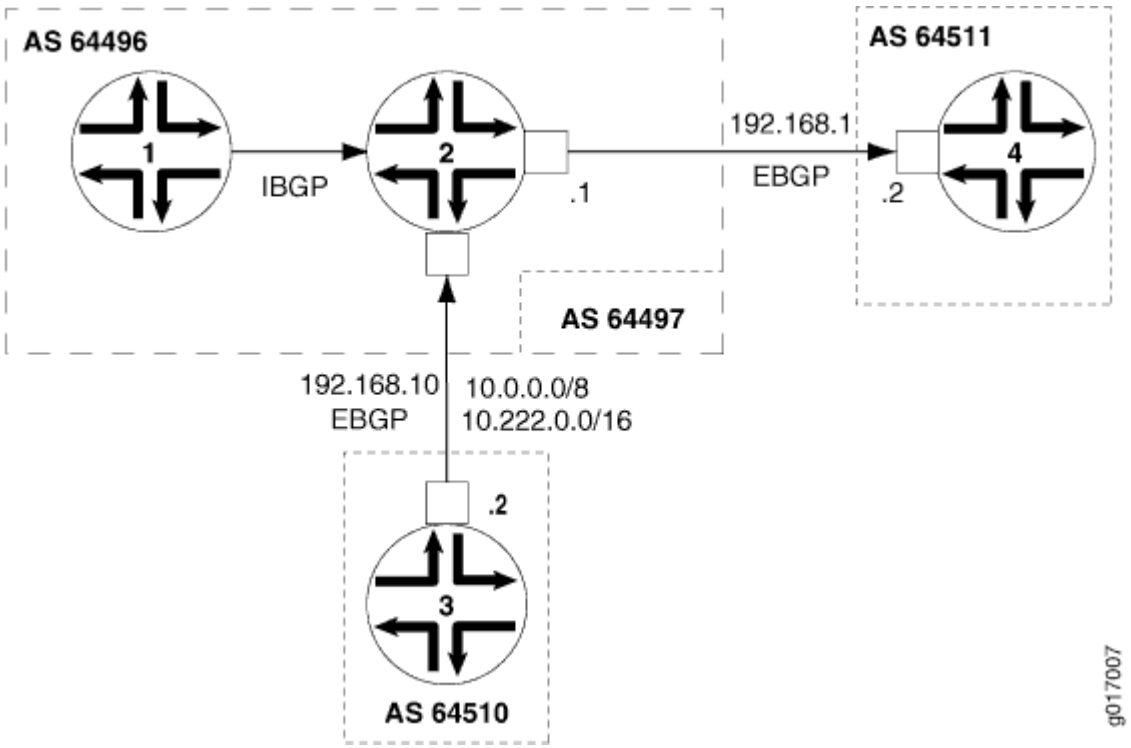
- **Local AS with private option**—When you use the `private` option, the local AS is used during the establishment of the BGP session with an EBGP neighbor but is hidden in the AS path sent to other IBGP and EBGP peers. Only the global AS is included in the AS path sent to external peers.

The `private` option is useful for establishing local peering with routing devices that remain configured with their former AS or with a specific customer that has not yet modified its peer arrangements. The local AS is used to establish the BGP session with the EBGP neighbor but is hidden in the AS path sent to external peers in another AS.

Include the `private` option so that the local AS is not prepended before the global AS in the AS path sent to external peers. When you specify the `private` option, the local AS is prepended only in the AS path sent to the EBGP neighbor.

For example, in Figure 9 on page 144, Router 1 and Router 2 are in AS 64496, Router 4 is in AS 64511, and Router 3 is in AS 64510. Router 2 formerly belonged to AS 64497, which has merged with another network and now belongs to AS 64496. Because Router 3 still peers with Router 2 using its former AS (64497), Router 2 needs to be configured with a local AS of 64497 in order to maintain peering with Router 3. Configuring a local AS of 64497 permits Router 2 to add AS 64497 when advertising routes to Router 3. Router 3 sees an AS path of 64497 64496 for the prefix 10/8.

Figure 9: Local AS Configuration



To prevent Router 2 from adding the local AS number in its announcements to other peers, use the `local-as 64497 private` statement. This statement configures Router 2 to not include local AS 64497 when announcing routes to Router 1 and to Router 4. In this case, Router 4 sees an AS path of 64496 64510 for the prefix 10.222/16.

- **Local AS with alias option**—In Junos OS Release 9.5 and later, you can configure a local AS as an alias. During the establishment of the BGP open session, the AS used in the open message alternates between the local AS and the global AS. If the local AS is used to connect with the EBGP neighbor, then only the local AS is prepended to the AS path when the BGP peer session is established. If the global AS is used to connect with the EBGP neighbor, then only the global AS is prepended to the AS path when the BGP peer session is established. The use of the `alias` option also means that the local AS is not prepended to the AS path for any routes learned from that EBGP neighbor. Therefore, the local AS remains hidden from other external peers.

  Configuring a local AS with the `alias` option is especially useful when you are migrating the routing devices in an acquired network to the new AS. During the migration process, some routing devices might be configured with the new AS while others remain configured with the former AS. For example, it is good practice to start by first migrating to the new AS any routing devices that function as route reflectors. However, as you migrate the route reflector clients incrementally, each route reflector has to peer with routing devices configured with the former AS, as well as peer with routing devices configured with the new AS. To establish local peer sessions, it can be useful for the BGP peers in the network to use both the local AS and the global AS. At the same time, you want to hide this local AS from external peers and use only the global AS in the AS path when exporting routes to another AS. In this kind of situation, configure the `alias` option.

  Include the `alias` option to configure the local AS as an alias to the global AS configured at the `[edit routing-options]` hierarchy level. When you configure a local AS as an alias, during the establishment of the BGP open session, the AS used in the open message alternates between the local AS and the global AS. The local AS is prepended to the AS path only when the peer session with an EBGP neighbor is established using that local AS. The local AS is hidden in the AS path sent to any other external peers. Only the global AS is prepended to the AS path when the BGP session is established using the global AS.

  NOTE: The `private` and `alias` options are mutually exclusive. You cannot configure both options with the same `local-as` statement.

- **Local AS with option not to prepend the global AS**—In Junos OS Release 9.6 and later, you can configure a local AS with the option not to prepend the global AS. Only the local AS is included in the AS path sent to external peers.

  Use the `no-prepend-global-as` option when you want to strip the global AS number from outbound BGP updates in a virtual private network (VPN) scenario. This option is useful in aVPN scenario in which you want to hide the global AS from the VPN.

  Include the `no-prepend-global-as` option to have the global AS configured at the `[edit routing-options]` hierarchy level removed from the AS path sent to external peers. When you use this option, only the local AS is included in the AS path for the routes sent to a customer edge (CE) device.

- **Number of loops option**—The local AS feature also supports specifying the number of times that detection of the AS number in the AS_PATH attribute causes the route to be discarded or hidden. For example, if you configure `loops 1`, the route is hidden if the AS number is detected in the path one or more times. This is the default behavior. If you configure `loops 2`, the route is hidden if the AS number is detected in the path two or more times.

  For the `loops` *`number`* statement, you can configure 1 through 10.

  NOTE: If you configure the local AS values for any BGP group, the detection of routing loops is performed using both the AS and the local AS values for all BGP groups.

  If the local AS for the EBGP or IBGP peer is the same as the current AS, do not use the `local-as` statement to specify the local AS number.

  When you configure the local AS within a VRF, this impacts the AS path loop-detection mechanism. All of the `local-as` statements configured on the device are part of a single AS domain. The AS path loop-detection mechanism is based on looking for a matching AS present in the domain.

SEE ALSO

Example: Configuring a Private Local AS for EBGP Sessions | **162**

Example: Configuring a Local AS for EBGP Sessions

IN THIS SECTION

- Requirements | **147**
- Overview | **147**
- Configuration | **148**
- Verification | **157**

This example shows how to configure a local autonomous system (AS) for a BGP peer so that both the global AS and the local AS are used in BGP inbound and outbound updates.

Requirements

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

Overview

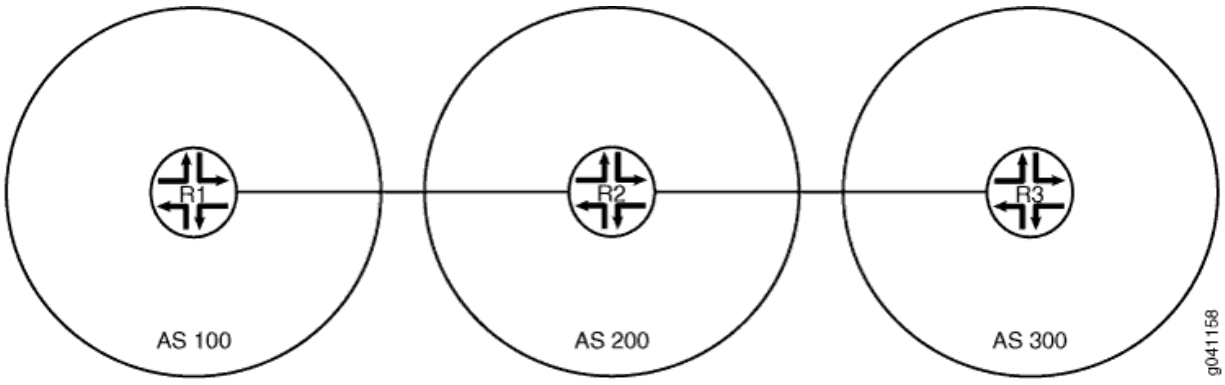
Use the `local-as` statement when ISPs merge and want to preserve a customer's configuration, particularly the AS with which the customer is configured to establish a peer relationship. The `local-as` statement simulates the AS number already in place in customer routers, even if the ISP's router has moved to a different AS.

This example shows how to use the `local-as` statement to configure a local AS. The `local-as` statement is supported for BGP at the global, group, and neighbor hierarchy levels.

When you configure the `local-as` statement, you must specify an AS number. You can specify a number from 1 through 4,294,967,295 in plain-number format. In Junos OS Release 9.1 and later, the range for AS numbers is extended to provide BGP support for 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*. In Junos OS Release 9.3 and later, you can also configure a 4-byte AS number using the AS-dot notation format of two integer values joined by a period: <*16-bit high-order value in decimal*>.<*16-bit low-order value in decimal*>. For example, the 4-byte AS number of 65,546 in plain-number format is represented as 1.10 in the AS-dot notation format. You can specify a value from 0.0 through 65535.65535 in AS-dot notation format. Junos OS continues to support 2-byte AS numbers. The 2-byte AS number range is 1 through 65,535 (this is a subset of the 4-byte range).

Figure 10 on page 147 shows the sample topology.

Figure 10: Topology for Configuring the Local AS



In this example, Device R2 formerly belonged to AS 250 and now is in AS 200. Device R1 and Device R3 are configured to peer with AS 250 instead of with the new AS number (AS 200). Device R2 has the new AS number configured with the `autonomous-system 200` statement. To enable the peering sessions to work, the `local-as 250` statement is added in the BGP configuration. Because `local-as 250` is configured, Device R2 includes both the global AS (200) and the local AS (250) in its BGP inbound and outbound updates.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 148
- Configuring Device R1 | 149
- Configuring Device R2 | 152
- Configuring Device R3 | 155

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 250
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.1.0.0/30 next-hop 10.0.0.2
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 3 family inet address 10.1.0.1/30
set interfaces lo0 unit 2 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
```

```
set protocols bgp group ext export send-static
set protocols bgp group ext local-as 250
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/0 unit 4 family inet address 10.1.0.2/30
set interfaces lo0 unit 3 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 250
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.0.0.0/30 next-hop 10.1.0.1
set routing-options autonomous-system 300
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
user@R1# set lo0 unit 1 family inet address 192.168.0.1/32
```

2. Configure external BGP (EBGP).

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set export send-direct
user@R1# set export send-static
user@R1# set peer-as 250
user@R1# set neighbor 10.0.0.2
```

3. Configure the routing policy.

```
[edit policy-options]
user@R1# set policy-statement send-direct term 1 from protocol direct
user@R1# set policy-statement send-direct term 1 then accept
user@R1# set policy-statement send-static term 1 from protocol static
user@R1# set policy-statement send-static term 1 then accept
```

4. Configure a static route to the remote network between Device R2 and Device R3.

```
[edit routing-options]
user@R1# set static route 10.1.0.0/30 next-hop 10.0.0.2
```

5. Configure the global AS number.

```
[edit routing-options]
user@R1# set autonomous-system 100
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@R1# show protocols
bgp {
    group ext {
```

```
        type external;
        export [ send-direct send-static ];
        peer-as 250;
        neighbor 10.0.0.2;
    }
}
```

```
user@R1# show routing-options
static {
    route 10.1.0.0/30 next-hop 10.0.0.2;
}
autonomous-system 100;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 2 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 3 family inet address 10.1.0.1/30
user@R2# set lo0 unit 2 family inet address 192.168.0.2/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set export send-direct
user@R2# set export send-static
```

```
user@R2# set neighbor 10.0.0.1 peer-as 100
user@R2# set neighbor 10.1.0.2 peer-as 300
```

3. Configure the local autonomous system (AS) number.

```
[edit protocols bgp group ext]
user@R2# set local-as 250
```

4. Configure the global AS number.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

5. Configure the routing policy.

```
[edit policy-options]
user@R2# set policy-statement send-direct term 1 from protocol direct
user@R2# set policy-statement send-direct term 1 then accept
user@R2# set policy-statement send-static term 1 from protocol static
user@R2# set policy-statement send-static term 1 then accept
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 3 {
        family inet {
```

```
            address 10.1.0.1/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        type external;
        export [ send-direct send-static ];
        local-as 250;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
        neighbor 10.1.0.2 {
            peer-as 300;
        }
```

```
    }
}
```

```
user@R2# show routing-options
autonomous-system 200;
```

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

Configuring Device R3

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 Junos OS CLI User Guide.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces]
user@R3# set fe-1/2/0 unit 4 family inet address 10.1.0.2/30
user@R3# set lo0 unit 3 family inet address 192.168.0.3/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@R3# set type external
user@R3# set export send-direct
user@R3# set export send-static
user@R3# set peer-as 250
user@R3# set neighbor 10.1.0.1
```

3. Configure the global autonomous system (AS) number.

```
[edit routing-options]
user@R3# set autonomous-system 300
```

4. Configure a static route to the remote network between Device R1 and Device R2.

```
[edit routing-options]
user@R3# set static route 10.0.0.0/30 next-hop 10.1.0.1
```

5. Configure the routing policy.

```
[edit policy-options]
user@R3# set policy-statement send-direct term 1 from protocol direct
user@R3# set policy-statement send-direct term 1 then accept
user@R3# set policy-statement send-static term 1 from protocol static
user@R3# set policy-statement send-static term 1 then accept
```

Results

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

```
user@R3# show interfaces
fe-1/2/0 {
    unit 4 {
        family inet {
            address 10.1.0.2/30;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.0.3/32;
        }
    }
}
```

```
user@R3# show policy-options
policy-statement send-direct {
    term 1 {
```

```
        from protocol direct;
        then accept;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@R3# show protocols
bgp {
    group ext {
        type external;
        export [ send-direct send-static ];
        peer-as 250;
        neighbor 10.1.0.1;
    }
}
```

```
user@R3# show routing-options
static {
    route 10.0.0.0/30 next-hop 10.1.0.1;
}
autonomous-system 300;
```

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

Verification

IN THIS SECTION

- Checking the Local and Global AS Settings | **158**
- Checking the BGP Peering Sessions | **160**
- Verifying the BGP AS Paths | **161**

Confirm that the configuration is working properly.

Checking the Local and Global AS Settings

Purpose

Make sure that Device R2 has the local and global AS settings configured.

Action

From operational mode, enter the `show bgp neighbors` command.

```
user@R2> show bgp neighbors
Peer: 10.0.0.1+179 AS 100      Local: 10.0.0.2+61036 AS 250
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct send-static ]
  Options: <Preference PeerAS LocalAS Refresh>
  Holdtime: 90 Preference: 170 Local AS: 250 Local System AS: 200
  Number of flaps: 0
  Peer ID: 192.168.0.1     Local ID: 192.168.0.2       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/2/0.2
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 100)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            3
    Accepted prefixes:            2
```

```
    Suppressed due to damping:    0
    Advertised prefixes:          4
  Last traffic (seconds): Received 6    Sent 14   Checked 47
  Input messages:  Total 258    Updates 3       Refreshes 0     Octets 4969
  Output messages: Total 258    Updates 2       Refreshes 0     Octets 5037
  Output Queue[0]: 0

Peer: 10.1.0.2+179 AS 300      Local: 10.1.0.1+52296 AS 250
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct send-static ]
  Options: <Preference PeerAS LocalAS Refresh>
  Holdtime: 90 Preference: 170 Local AS: 250 Local System AS: 200
  Number of flaps: 0
  Peer ID: 192.168.0.3     Local ID: 192.168.0.2       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  Local Interface: fe-1/2/1.3
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 300)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            3
    Accepted prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          4
  Last traffic (seconds): Received 19   Sent 26   Checked 9
  Input messages:  Total 256    Updates 3       Refreshes 0     Octets 4931
  Output messages: Total 256    Updates 2       Refreshes 0     Octets 4999
  Output Queue[0]: 0
```

Meaning

The Local AS: 250 and Local System AS: 200 output shows that Device R2 has the expected settings. Additionally, the output shows that the options list includes LocalAS.

Checking the BGP Peering Sessions

Purpose

Ensure that the sessions are established and that the local AS number 250 is displayed.

Action

From operational mode, enter the `show bgp summary` command.

```
user@R1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 4          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.0.0.2                250        232        233       0       4     1:42:37
2/4/4/0              0/0/0/0
```

```
user@R3> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 4          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.1.0.1                250        235        236       0       4     1:44:25
2/4/4/0              0/0/0/0
```

Meaning

Device R1 and Device R3 appear to be peering with a device in AS 250, even though Device R2 is actually in AS 200.

Verifying the BGP AS Paths

Purpose

Make sure that the routes are in the routing tables and that the AS paths show the local AS number 250.

Action

From configuration mode, enter the `set route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 6 destinations, 8 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/30         [BGP/170] 01:46:44, localpref 100
                      AS path: 250 I
                    > to 10.0.0.2 via fe-1/2/0.1
10.1.0.0/30         [BGP/170] 01:46:44, localpref 100
                      AS path: 250 I
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.0.2/32     *[BGP/170] 01:46:44, localpref 100
                      AS path: 250 I
                     > to 10.0.0.2 via fe-1/2/0.1
192.168.0.3/32     *[BGP/170] 01:46:40, localpref 100
                      AS path: 250 300 I
                     > to 10.0.0.2 via fe-1/2/0.1
```

```
user@R3> show route protocol bgp

inet.0: 6 destinations, 8 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/30         [BGP/170] 01:47:10, localpref 100
                      AS path: 250 I
                    > to 10.1.0.1 via fe-1/2/0.4
10.1.0.0/30         [BGP/170] 01:47:10, localpref 100
                      AS path: 250 I
                    > to 10.1.0.1 via fe-1/2/0.4
192.168.0.1/32     *[BGP/170] 01:47:10, localpref 100
                      AS path: 250 100 I
```

```
                    > to 10.1.0.1 via fe-1/2/0.4
192.168.0.2/32     *[BGP/170] 01:47:10, localpref 100
                      AS path: 250 I
                    > to 10.1.0.1 via fe-1/2/0.4
```

Meaning

The output shows that Device R1 and Device R3 appear to have routes with AS paths that include AS 250, even though Device R2 is actually in AS 200.

SEE ALSO

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

Example: Configuring a Private Local AS for EBGP Sessions

IN THIS SECTION

- Requirements | 162
- Overview | 163
- Configuration | 164
- Verification | 169

This example shows how to configure a private local autonomous system (AS) number. The local AS is considered to be private because it is advertised to peers that use the local AS number for peering, but is hidden in the announcements to peers that can use the global AS number for peering.

Requirements

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

Overview

Use the `local-as` statement when ISPs merge and want to preserve a customer's configuration, particularly the AS with which the customer is configured to establish a peer relationship. The `local-as` statement simulates the AS number already in place in customer routers, even if the ISP's router has moved to a different AS.

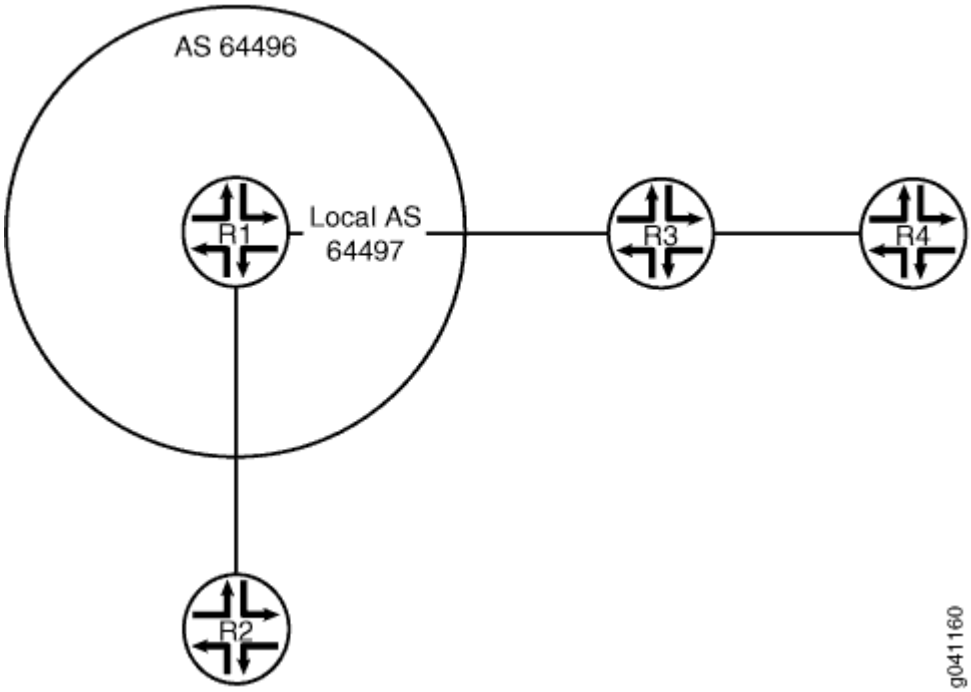
When you use the `private` option, the local AS is used during the establishment of the BGP session with an external BGP (EBGP) neighbor, but is hidden in the AS path sent to other EBGP peers. Only the global AS is included in the AS path sent to external peers.

The `private` option is useful for establishing local peering with routing devices that remain configured with their former AS or with a specific customer that has not yet modified its peer arrangements. The local AS is used to establish the BGP session with the EBGP neighbor, but is hidden in the AS path sent to external peers in another AS.

Include the `private` option so that the local AS is not prepended before the global AS in the AS path sent to external peers. When you specify the `private` option, the local AS is prepended only in the AS path sent to the EBGP neighbor.

Figure 11 on page 163 shows the sample topology.

Figure 11: Topology for Configuring a Private Local AS



Device R1 is in AS 64496. Device R2 is in AS 64510. Device R3 is in AS 64511. Device R4 is in AS 64512. Device R1 formerly belonged to AS 64497, which has merged with another network and now

belongs to AS 64496. Because Device R3 still peers with Device R1, using its former AS, 64497, Device R1 needs to be configured with a local AS of 64497 in order to maintain peering with Device R3. Configuring a local AS of 64497 permits Device R1 to add AS 64497 when advertising routes to Device R3. Device R3 sees an AS path of 64497 64496 for the prefix 10.1.1.2/32, which is Device R2's loopback interface. Device R4, which is behind Device R3, sees an AS path of 64511 64497 64496 64510 to Device R2's loopback interface. To prevent Device R1 from adding the local AS number in its announcements to other peers, this example includes the `local-as 64497 private` statement. The `private` option configures Device R1 to not include the local AS 64497 when announcing routes to Device R2. Device R2 sees an AS path of 64496 64511 to Device R3 and an AS path of 64496 64511 64512 to Device R4. The `private` option in Device R1's configuration causes the AS number 64497 to be missing from the AS paths that Device R1 readvertises to Device R2.

Device R1 is hiding the private local AS from all the routers, except Device R3. The `private` option applies to the routes that Device R1 receives (learns) from Device R3 and that Device R1, in turn, readvertises to other routers. When these routes learned from Device R3 are readavertised by Device R1 to Device R2, the private local AS is missing from the AS path advertised to Device R2.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **164**
- Configuring Device R1 | **166**

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 R1

```
set interfaces fe-1/2/0 unit 3 family inet address 192.168.1.1/24
set interfaces fe-1/2/1 unit 5 family inet address 192.168.10.1/24
set interfaces lo0 unit 2 family inet address 10.1.1.1/32
set protocols bgp group external-AS64511 type external
set protocols bgp group external-AS64511 peer-as 64511
set protocols bgp group external-AS64511 local-as 64497
set protocols bgp group external-AS64511 local-as private
set protocols bgp group external-AS64511 neighbor 192.168.1.2
```

```
set protocols bgp group external-AS64510 type external
set protocols bgp group external-AS64510 peer-as 64510
set protocols bgp group external-AS64510 neighbor 192.168.10.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64496
```

Device R2

```
set interfaces fe-1/2/0 unit 6 family inet address 192.168.10.2/24
set interfaces lo0 unit 3 family inet address 10.1.1.2/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 64496
set protocols bgp group external neighbor 192.168.10.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/0 unit 4 family inet address 192.168.1.2/24
set interfaces fe-1/2/1 unit 7 family inet address 192.168.5.1/24
set interfaces lo0 unit 4 family inet address 10.1.1.3/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external neighbor 192.168.1.1 peer-as 64497
set protocols bgp group external neighbor 192.168.5.2 peer-as 64512
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64511
```

Device R4

```
set interfaces fe-1/2/0 unit 8 family inet address 192.168.5.2/24
set interfaces lo0 unit 5 family inet address 10.1.1.4/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 64511
set protocols bgp group external neighbor 192.168.5.1
set policy-options policy-statement send-direct term 1 from protocol direct
```

```
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64512
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 3]
user@R1# set family inet address 192.168.1.1/24
[edit interfaces fe-1/2/1 unit 5]
user@R1# set family inet address 192.168.10.1/24
[edit interfaces lo0 unit 2]
user@R1# set family inet address 10.1.1.1/32
```

2. Configure the EBGP peering session with Device R2.

```
[edit protocols bgp group external-AS64510]
user@R1# set type external
user@R1# set peer-as 64510
user@R1# set neighbor 192.168.10.2
```

3. Configure the EBGP peering session with Device R3.

```
[edit protocols bgp group external-AS64511]
user@R1# set type external
user@R1# set peer-as 64511
user@R1# set local-as 64497
user@R1# set local-as private
user@R1# set neighbor 192.168.1.2
```

4. Configure the routing policy.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

5. Configure the global autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 64496
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 3 {
        family inet {
            address 192.168.1.1/24;
        }
    }
}
fe-1/2/1 {
    unit 5 {
        family inet {
            address 192.168.10.1/24;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 10.1.1.1/32;
        }
```

```
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show protocols
bgp {
    group external-AS64511 {
        type external;
        peer-as 64511;
        local-as 64497 private;
        neighbor 192.168.1.2;
    }
    group external-AS64510 {
        type external;
        peer-as 64510;
        neighbor 192.168.10.2;
    }
}
```

```
user@R1# show routing-options
autonomous-system 64496;
```

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

Repeat the configuration as needed for the other devices in the topology.

Verification

IN THIS SECTION

- Checking Device R2's AS Paths | 169
- Checking Device R3's AS Paths | 170

Confirm that the configuration is working properly.

Checking Device R2's AS Paths

Purpose

Make sure that Device R2 does not have AS 64497 in its AS paths to Device R3 and Device R4.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R2> show route protocol bgp
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.3/32        *[BGP/170] 01:33:11, localpref 100
                      AS path: 64496 64511 I
                    > to 192.168.10.1 via fe-1/2/0.6
10.1.1.4/32        *[BGP/170] 01:33:11, localpref 100
                      AS path: 64496 64511 64512 I
                    > to 192.168.10.1 via fe-1/2/0.6
192.168.5.0/24     *[BGP/170] 01:49:15, localpref 100
                      AS path: 64496 64511 I
                    > to 192.168.10.1 via fe-1/2/0.6
```

Meaning

Device R2's AS paths do not include AS 64497.

Checking Device R3's AS Paths

Purpose

Make sure that the local AS 64497 is prepended only in the AS path sent to the EBGP neighbor R3 . Device R3 sees an AS path of 64497 64496 for the prefix 10.1.1.2/32, which is Device R2's loopback interface.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R3> show route protocol bgp
inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.2/32        *[BGP/170] 01:35:11, localpref 100
                      AS path: 64497 64496 64510 I
                    > to 192.168.1.1 via fe-1/2/0.4
10.1.1.4/32        *[BGP/170] 01:35:11, localpref 100
                      AS path: 64512 I
                    > to 192.168.5.2 via fe-1/2/1.7
192.168.5.0/24      [BGP/170] 01:51:15, localpref 100
                      AS path: 64512 I
                    > to 192.168.5.2 via fe-1/2/1.7
```

Meaning

Device R3's route to Device R2 (prefix 10.1.1.2) includes both the local and the global AS configured on Device R1 (64497 and 64496, respectively).

SEE ALSO

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

Understanding the Accumulated IGP Attribute for BGP

The interior gateway protocols (IGPs) are designed to handle routing within a single domain or an autonomous system (AS). Each link is assigned a particular value called a metric. The distance between the two nodes is calculated as a sum of all the metric values of links along the path. The IGP selects the shortest path between two nodes based on distance.

BGP is designed to provide routing over a large number of independent ASs with limited or no coordination among respective administrations. BGP does not use metrics in the path selection decisions.

The accumulated IGP (AIGP) metric attribute for BGP enables deployment in which a single administration can run several contiguous BGP ASs. Such deployments allow BGP to make routing decisions based on the IGP metric. In such networks, it is possible for BGP to select paths based on metrics as is done by IGPs. In this case, BGP chooses the shortest path between two nodes, even though the nodes might be in two different ASs.

The AIGP attribute is particularly useful in networks that use tunneling to deliver a packet to its BGP next hop. The Juniper Networks® Junos® operating system (Junos OS) currently supports the AIGP attribute for two BGP address families, `family inet labeled-unicast` and `family inet6 labeled-unicast`.

AIGP impacts the BGP best-route decision process. The AIGP attribute preference rule is applied after the local-preference rule. The AIGP distance is compared to break a tie. The BGP best-route decision process also impacts the way the interior cost rule is applied if the resolving next hop has an AIGP attribute. Without AIGP enabled, the interior cost of a route is based on the calculation of the metric to the next hop for the route. With AIGP enabled, the resolving AIGP distance is added to the interior cost.

Starting in Release 20.2R1, Junos OS supports the translation of AIGP metric to MED. You can enable this feature when you want the MED to carry the end to end AIGP metric value, which is used to choose the best path. This is especially useful in Inter-AS MPLS VPNs solution, where customer sites are connected via two different service providers, and customer edge routers want to take IGP metric based decision. You can configure a `minimum-aigp` to prevent unnecessary update of route when effective-aigp changes past the previously known lowest value. Effective AIGP is the AIGP value advertised with the route plus the IGP cost to reach the nexthop. You can configure `effective-aigp` and `minimum-effective-aigp` statements at the `[edit protocols bgp group <group-name> metric-out]` and `[edit policy-options policy-statement <name> then metric]` hierarchy levels.

The AIGP attribute is an optional non-transitive BGP path attribute and is specified in Internet draft draft-ietf-idr-aigp-06, *The Accumulated IGP Metric Attribute for BGP*.

SEE ALSO

Understanding AS Override | 227

Example: Configuring the Accumulated IGP Attribute for BGP

IN THIS SECTION

- Requirements | 172
- Overview | 172
- Configuration | 174
- Verification | 217

This example shows how to configure the accumulated IGP (AIGP) metric attribute for BGP.

Requirements

This example uses the following hardware and software components:

- Seven BGP-speaking devices.
- Junos OS Release 12.1 or later.

Overview

IN THIS SECTION

- Topology Diagram | 173

The AIGP attribute enables deployments in which a single administration can run several contiguous BGP autonomous systems (ASs). Such deployments allow BGP to make routing decisions based on the IGP metric. With AIGP enabled, BGP can select paths based on IGP metrics. This enables BGP to choose the shortest path between two nodes, even though the nodes might be in different ASs. The AIGP attribute is particularly useful in networks that use tunneling to deliver a packet to its BGP next hop. This example shows AIGP configured with MPLS label-switched paths.

To enable AIGP, you include the `aigp` statement in the BGP configuration on a protocol family basis. Configuring AIGP on a particular family enables sending and receiving of the AIGP attribute on that family. By default, AIGP is disabled. An AIGP-disabled neighbor does not send an AIGP attribute and silently discards a received AIGP attribute.

Junos OS supports AIGP for `family inet labeled-unicast` and `family inet6 labeled-unicast`. The `aigp` statement can be configured for a given family at the global BGP, group, or neighbor level.

By default, the value of the AIGP attribute for a local prefix is zero. An AIGP-enabled neighbor can originate an AIGP attribute for a given prefix by export policy, using the `aigp-originate` policy action. The value of the AIGP attribute reflects the IGP distance to the prefix. Alternatively, you can specify a value, by using the `aigp-originate distance` *`distance`* policy action. The configurable range is 0 through 4,294,967,295. Only one node needs to originate an AIGP attribute. The AIGP attribute is retained and readvertised if the neighbors are AIGP enabled with the `aigp` statement in the BGP configuration.

The policy action to originate the AIGP attribute has the following requirements:

- Neighbor must be AIGP enabled.
- Policy must be applied as an export policy.
- Prefix must have no current AIGP attribute.
- Prefix must export with next-hop self.
- Prefix must reside within the AIGP domain. Typically, a loopback IP address is the prefix to originate.

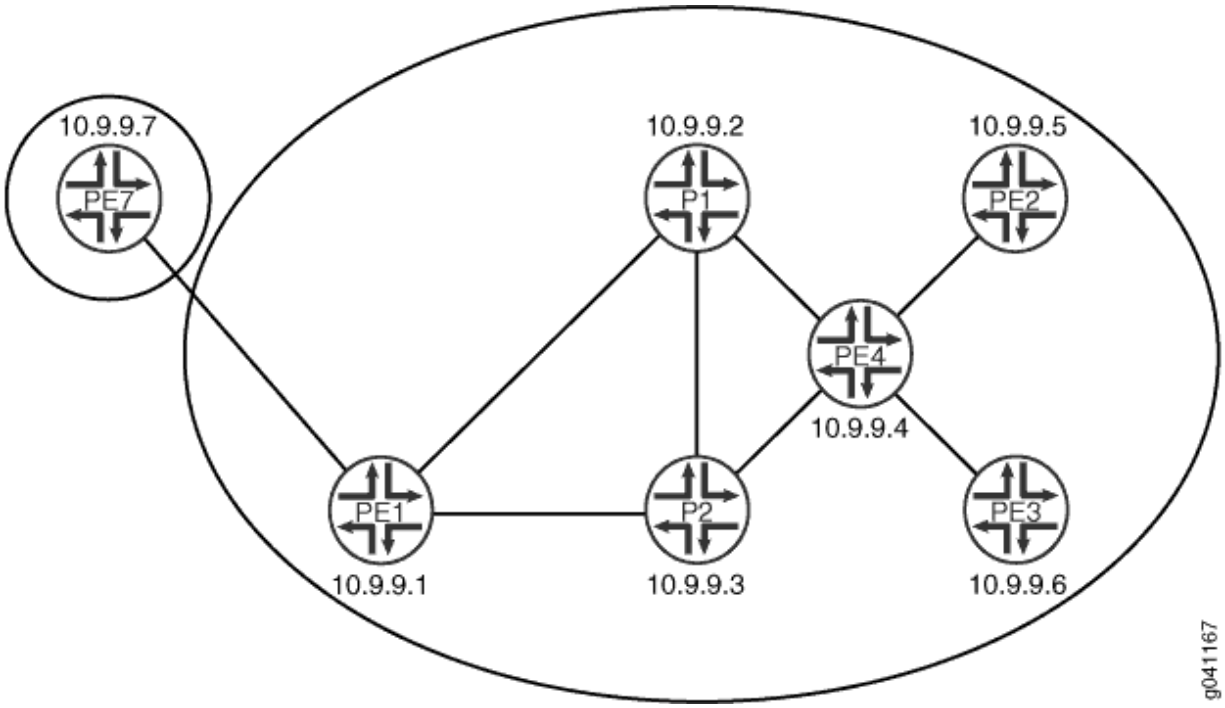
The policy is ignored if these requirements are not met.

Topology Diagram

Figure 12 on page 174 shows the topology used in this example. OSPF is used as the interior gateway protocol (IGP). Internal BGP (IBGP) is configured between Device PE1 and Device PE4. External BGP (EBGP) is configured between Device PE7 and Device PE1, between Device PE4 and Device PE3, and between Device PE4 and Device PE2. Devices PE4, PE2, and PE3 are configured for multihop. Device PE4 selects a path based on the AIGP value and then readvertises the AIGP value based on the AIGP and policy configuration. Device PE1 readvertises the AIGP value to Device PE7, which is in another administrative domain. Every device has two loopback interface addresses: 10.9.9.x is used for BGP peering and the router ID, and 10.100.1.x is used for the BGP next hop.

The network between Device PE1 and PE3 has IBGP peering and multiple OSPF areas. The external link to Device PE7 is configured to show that the AIGP attribute is readvertised to a neighbor outside of the administrative domain, if that neighbor is AIGP enabled.

Figure 12: Advertisement of Multiple Paths in BGP



For origination of an AIGP attribute, the BGP next hop is required to be itself. If the BGP next hop remains unchanged, the received AIGP attribute is readvertised, as is, to another AIGP neighbor. If the next hop changes, the received AIGP attribute is readvertised with an increased value to another AIGP neighbor. The increase in value reflects the IGP distance to the previous BGP next hop. To demonstrate, this example uses loopback interface addresses for Device PE4's EBGP peering sessions with Device PE2 and Device PE3. Multihop is enabled on these sessions so that a recursive lookup is performed to determine the point-to-point interface. Because the next hop changes, the IGP distance is added to the AIGP distance.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **175**
- Configuring Device P1 | **182**
- Configuring Device P2 | **187**
- Configuring Device PE4 | **191**
- Configuring Device PE1 | **199**
- Configuring Device PE2 | **204**

Configuring Device PE3 | 210
Configuring Device PE7 | 215

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 P1

```
set interfaces fe-1/2/0 unit 1 description P1-to-PE1
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.2/30
set interfaces fe-1/2/0 unit 1 family mpls
set interfaces fe-1/2/1 unit 4 description P1-to-P2
set interfaces fe-1/2/1 unit 4 family inet address 10.0.0.29/30
set interfaces fe-1/2/1 unit 4 family mpls
set interfaces fe-1/2/2 unit 8 description P1-to-PE4
set interfaces fe-1/2/2 unit 8 family inet address 10.0.0.17/30
set interfaces fe-1/2/2 unit 8 family mpls
set interfaces lo0 unit 3 family inet address 10.9.9.2/32
set interfaces lo0 unit 3 family inet address 10.100.1.2/32
set protocols rsvp interface fe-1/2/0.1
set protocols rsvp interface fe-1/2/2.8
set protocols rsvp interface fe-1/2/1.4
set protocols mpls label-switched-path P1-to-P2 to 10.9.9.3
set protocols mpls label-switched-path P1-to-PE1 to 10.9.9.1
set protocols mpls label-switched-path P1-to-PE4 to 10.9.9.4
set protocols mpls interface fe-1/2/0.1
set protocols mpls interface fe-1/2/2.8
set protocols mpls interface fe-1/2/1.4
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.9.9.2
set protocols bgp group internal family inet labeled-unicast aigp
set protocols bgp group internal neighbor 10.9.9.1
set protocols bgp group internal neighbor 10.9.9.3
set protocols bgp group internal neighbor 10.9.9.4
set protocols ospf area 0.0.0.1 interface fe-1/2/0.1 metric 1
set protocols ospf area 0.0.0.1 interface fe-1/2/1.4 metric 1
```

```
set protocols ospf area 0.0.0.0 interface fe-1/2/2.8 metric 1
set protocols ospf area 0.0.0.0 interface 10.9.9.2 passive
set protocols ospf area 0.0.0.0 interface 10.9.9.2 metric 1
set protocols ospf area 0.0.0.0 interface 10.100.1.2 passive
set protocols ospf area 0.0.0.0 interface 10.100.1.2 metric 1
set routing-options router-id 10.9.9.2
set routing-options autonomous-system 13979
```

Device P2

```
set interfaces fe-1/2/0 unit 3 description P2-to-PE1
set interfaces fe-1/2/0 unit 3 family inet address 10.0.0.6/30
set interfaces fe-1/2/0 unit 3 family mpls
set interfaces fe-1/2/1 unit 5 description P2-to-P1
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.30/30
set interfaces fe-1/2/1 unit 5 family mpls
set interfaces fe-1/2/2 unit 6 description P2-to-PE4
set interfaces fe-1/2/2 unit 6 family inet address 10.0.0.13/30
set interfaces fe-1/2/2 unit 6 family mpls
set interfaces lo0 unit 5 family inet address 10.9.9.3/32
set interfaces lo0 unit 5 family inet address 10.100.1.3/32
set protocols rsvp interface fe-1/2/1.5
set protocols rsvp interface fe-1/2/2.6
set protocols rsvp interface fe-1/2/0.3
set protocols mpls label-switched-path P2-to-PE1 to 10.9.9.1
set protocols mpls label-switched-path P2-to-P1 to 10.9.9.2
set protocols mpls label-switched-path P2-to-PE4 to 10.9.9.4
set protocols mpls interface fe-1/2/1.5
set protocols mpls interface fe-1/2/2.6
set protocols mpls interface fe-1/2/0.3
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.9.9.3
set protocols bgp group internal family inet labeled-unicast aigp
set protocols bgp group internal neighbor 10.9.9.1
set protocols bgp group internal neighbor 10.9.9.2
set protocols bgp group internal neighbor 10.9.9.4
set protocols ospf area 0.0.0.0 interface fe-1/2/2.6 metric 1
set protocols ospf area 0.0.0.0 interface 10.9.9.3 passive
set protocols ospf area 0.0.0.0 interface 10.9.9.3 metric 1
set protocols ospf area 0.0.0.0 interface 10.100.1.3 passive
set protocols ospf area 0.0.0.0 interface 10.100.1.3 metric 1
```

```
set routing-options router-id 10.9.9.3
set routing-options autonomous-system 13979
```

Device PE4

```
set interfaces fe-1/2/0 unit 7 description PE4-to-P2
set interfaces fe-1/2/0 unit 7 family inet address 10.0.0.14/30
set interfaces fe-1/2/0 unit 7 family mpls
set interfaces fe-1/2/1 unit 9 description PE4-to-P1
set interfaces fe-1/2/1 unit 9 family inet address 10.0.0.18/30
set interfaces fe-1/2/1 unit 9 family mpls
set interfaces fe-1/2/2 unit 10 description PE4-to-PE2
set interfaces fe-1/2/2 unit 10 family inet address 10.0.0.21/30
set interfaces fe-1/2/2 unit 10 family mpls
set interfaces fe-1/0/2 unit 12 description PE4-to-PE3
set interfaces fe-1/0/2 unit 12 family inet address 10.0.0.25/30
set interfaces fe-1/0/2 unit 12 family mpls
set interfaces lo0 unit 7 family inet address 10.9.9.4/32
set interfaces lo0 unit 7 family inet address 10.100.1.4/32
set protocols rsvp interface fe-1/2/0.7
set protocols rsvp interface fe-1/2/1.9
set protocols rsvp interface fe-1/2/2.10
set protocols rsvp interface fe-1/0/2.12
set protocols mpls label-switched-path PE4-to-PE2 to 10.9.9.5
set protocols mpls label-switched-path PE4-to-PE3 to 10.9.9.6
set protocols mpls label-switched-path PE4-to-P1 to 10.9.9.2
set protocols mpls label-switched-path PE4-to-P2 to 10.9.9.3
set protocols mpls interface fe-1/2/0.7
set protocols mpls interface fe-1/2/1.9
set protocols mpls interface fe-1/2/2.10
set protocols mpls interface fe-1/0/2.12
set protocols bgp export next-hop
set protocols bgp export aigp
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.9.9.4
set protocols bgp group internal family inet labeled-unicast aigp
set protocols bgp group internal neighbor 10.9.9.1
set protocols bgp group internal neighbor 10.9.9.3
set protocols bgp group internal neighbor 10.9.9.2
set protocols bgp group external type external
set protocols bgp group external multihop ttl 2
set protocols bgp group external local-address 10.9.9.4
```

```
set protocols bgp group external family inet labeled-unicast aigp
set protocols bgp group external peer-as 7018
set protocols bgp group external neighbor 10.9.9.5
set protocols bgp group external neighbor 10.9.9.6
set protocols ospf area 0.0.0.0 interface fe-1/2/1.9 metric 1
set protocols ospf area 0.0.0.0 interface fe-1/2/0.7 metric 1
set protocols ospf area 0.0.0.0 interface 10.9.9.4 passive
set protocols ospf area 0.0.0.0 interface 10.9.9.4 metric 1
set protocols ospf area 0.0.0.0 interface 10.100.1.4 passive
set protocols ospf area 0.0.0.0 interface 10.100.1.4 metric 1
set protocols ospf area 0.0.0.2 interface fe-1/2/2.10 metric 1
set protocols ospf area 0.0.0.3 interface fe-1/0/2.12 metric 1
set policy-options policy-statement aigp term 10 from protocol static
set policy-options policy-statement aigp term 10 from route-filter 44.0.0.0/24 exact
set policy-options policy-statement aigp term 10 then aigp-originate distance 200
set policy-options policy-statement aigp term 10 then next-hop 10.100.1.4
set policy-options policy-statement aigp term 10 then accept
set policy-options policy-statement next-hop term 10 from protocol bgp
set policy-options policy-statement next-hop term 10 then next-hop 10.100.1.4
set policy-options policy-statement next-hop term 10 then accept
set policy-options policy-statement next-hop term 20 from protocol direct
set policy-options policy-statement next-hop term 20 from route-filter 10.9.9.4/32 exact
set policy-options policy-statement next-hop term 20 from route-filter 10.100.1.4/32 exact
set policy-options policy-statement next-hop term 20 then next-hop 10.100.1.4
set policy-options policy-statement next-hop term 20 then accept
set routing-options static route 44.0.0.0/24 discard
set routing-options router-id 10.9.9.4
set routing-options autonomous-system 13979
```

Device PE1

```
set interfaces fe-1/2/0 unit 0 description PE1-to-P1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 2 description PE1-to-P2
set interfaces fe-1/2/1 unit 2 family inet address 10.0.0.5/30
set interfaces fe-1/2/1 unit 2 family mpls
set interfaces fe-1/2/2 unit 14 description PE1-to-PE7
set interfaces fe-1/2/2 unit 14 family inet address 10.0.0.9/30
set interfaces lo0 unit 1 family inet address 10.9.9.1/32
set interfaces lo0 unit 1 family inet address 10.100.1.1/32
set protocols rsvp interface fe-1/2/0.0
```

```
set protocols rsvp interface fe-1/2/1.2
set protocols rsvp interface fe-1/2/2.14
set protocols mpls label-switched-path PE1-to-P1 to 10.9.9.2
set protocols mpls label-switched-path PE1-to-P2 to 10.9.9.3
set protocols mpls interface fe-1/2/0.0
set protocols mpls interface fe-1/2/1.2
set protocols mpls interface fe-1/2/2.14
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.9.9.1
set protocols bgp group internal family inet labeled-unicast aigp
set protocols bgp group internal export SET_EXPORT_ROUTES
set protocols bgp group internal vpn-apply-export
set protocols bgp group internal neighbor 10.9.9.4
set protocols bgp group internal neighbor 10.9.9.2
set protocols bgp group internal neighbor 10.9.9.3
set protocols bgp group external type external
set protocols bgp group external family inet labeled-unicast aigp
set protocols bgp group external export SET_EXPORT_ROUTES
set protocols bgp group external peer-as 7019
set protocols bgp group external neighbor 10.0.0.10
set protocols ospf area 0.0.0.1 interface fe-1/2/0.0 metric 1
set protocols ospf area 0.0.0.1 interface fe-1/2/1.2 metric 1
set protocols ospf area 0.0.0.1 interface 10.9.9.1 passive
set protocols ospf area 0.0.0.1 interface 10.9.9.1 metric 1
set protocols ospf area 0.0.0.1 interface 10.100.1.1 passive
set protocols ospf area 0.0.0.1 interface 10.100.1.1 metric 1
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.1
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then accept
set routing-options router-id 10.9.9.1
set routing-options autonomous-system 13979
```

Device PE2

```
set interfaces fe-1/2/0 unit 11 description PE2-to-PE4
set interfaces fe-1/2/0 unit 11 family inet address 10.0.0.22/30
set interfaces fe-1/2/0 unit 11 family mpls
set interfaces lo0 unit 9 family inet address 10.9.9.5/32 primary
set interfaces lo0 unit 9 family inet address 10.100.1.5/32
set protocols rsvp interface fe-1/2/0.11
set protocols mpls label-switched-path PE2-to-PE4 to 10.9.9.4
```

```
set protocols mpls interface fe-1/2/0.11
set protocols bgp group external type external
set protocols bgp group external multihop ttl 2
set protocols bgp group external local-address 10.9.9.5
set protocols bgp group external family inet labeled-unicast aigp
set protocols bgp group external export next-hop
set protocols bgp group external export aigp
set protocols bgp group external export SET_EXPORT_ROUTES
set protocols bgp group external vpn-apply-export
set protocols bgp group external peer-as 13979
set protocols bgp group external neighbor 10.9.9.4
set protocols ospf area 0.0.0.2 interface 10.9.9.5 passive
set protocols ospf area 0.0.0.2 interface 10.9.9.5 metric 1
set protocols ospf area 0.0.0.2 interface 10.100.1.5 passive
set protocols ospf area 0.0.0.2 interface 10.100.1.5 metric 1
set protocols ospf area 0.0.0.2 interface fe-1/2/0.11 metric 1
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol static
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.5
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then accept
set policy-options policy-statement aigp term 10 from route-filter 55.0.0.0/24 exact
set policy-options policy-statement aigp term 10 then aigp-originate distance 20
set policy-options policy-statement aigp term 10 then next-hop 10.100.1.5
set policy-options policy-statement aigp term 10 then accept
set policy-options policy-statement aigp term 20 from route-filter 99.0.0.0/24 exact
set policy-options policy-statement aigp term 20 then aigp-originate distance 30
set policy-options policy-statement aigp term 20 then next-hop 10.100.1.5
set policy-options policy-statement aigp term 20 then accept
set policy-options policy-statement next-hop term 10 from protocol bgp
set policy-options policy-statement next-hop term 10 then next-hop 10.100.1.5
set policy-options policy-statement next-hop term 10 then accept
set policy-options policy-statement next-hop term 20 from protocol direct
set policy-options policy-statement next-hop term 20 from route-filter 10.9.9.5/32 exact
set policy-options policy-statement next-hop term 20 from route-filter 10.100.1.5/32 exact
set policy-options policy-statement next-hop term 20 then next-hop 10.100.1.5
set policy-options policy-statement next-hop term 20 then accept
set routing-options static route 99.0.0.0/24 discard
set routing-options static route 55.0.0.0/24 discard
set routing-options router-id 10.9.9.5
set routing-options autonomous-system 7018
```

Device PE3

```
set interfaces fe-1/2/0 unit 13 description PE3-to-PE4
set interfaces fe-1/2/0 unit 13 family inet address 10.0.0.26/30
set interfaces fe-1/2/0 unit 13 family mpls
set interfaces lo0 unit 11 family inet address 10.9.9.6/32
set interfaces lo0 unit 11 family inet address 10.100.1.6/32
set protocols rsvp interface fe-1/2/0.13
set protocols mpls label-switched-path PE3-to-PE4 to 10.9.9.4
set protocols mpls interface fe-1/2/0.13
set protocols bgp group external type external
set protocols bgp group external multihop ttl 2
set protocols bgp group external local-address 10.9.9.6
set protocols bgp group external family inet labeled-unicast aigp
set protocols bgp group external export next-hop
set protocols bgp group external export SET_EXPORT_ROUTES
set protocols bgp group external vpn-apply-export
set protocols bgp group external peer-as 13979
set protocols bgp group external neighbor 10.9.9.4
set protocols ospf area 0.0.0.3 interface 10.9.9.6 passive
set protocols ospf area 0.0.0.3 interface 10.9.9.6 metric 1
set protocols ospf area 0.0.0.3 interface 10.100.1.6 passive
set protocols ospf area 0.0.0.3 interface 10.100.1.6 metric 1
set protocols ospf area 0.0.0.3 interface fe-1/2/0.13 metric 1
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol static
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.6
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then accept
set policy-options policy-statement next-hop term 10 from protocol bgp
set policy-options policy-statement next-hop term 10 then next-hop 10.100.1.6
set policy-options policy-statement next-hop term 10 then accept
set policy-options policy-statement next-hop term 20 from protocol direct
set policy-options policy-statement next-hop term 20 from route-filter 10.9.9.6/32 exact
set policy-options policy-statement next-hop term 20 from route-filter 10.100.1.6/32 exact
set policy-options policy-statement next-hop term 20 then next-hop 10.100.1.6
set policy-options policy-statement next-hop term 20 then accept
set routing-options router-id 10.9.9.6
set routing-options autonomous-system 7018
```

Device PE7

```
set interfaces fe-1/2/0 unit 15 description PE7-to-PE1
set interfaces fe-1/2/0 unit 15 family inet address 10.0.0.10/30
set interfaces lo0 unit 13 family inet address 10.9.9.7/32
set interfaces lo0 unit 13 family inet address 10.100.1.7/32
set protocols bgp group external type external
set protocols bgp group external family inet labeled-unicast aigp
set protocols bgp group external export SET_EXPORT_ROUTES
set protocols bgp group external peer-as 13979
set protocols bgp group external neighbor 10.0.0.9
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
set policy-options policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.7
set policy-options policy-statement SET_EXPORT_ROUTES term 10 then accept
set routing-options router-id 10.9.9.7
set routing-options autonomous-system 7019
```

Configuring Device P1

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 Junos OS CLI User Guide.

To configure Device P1:

1. Configure the interfaces.

```
[edit interfaces]
user@P1# set fe-1/2/0 unit 1 description P1-to-PE1
user@P1# set fe-1/2/0 unit 1 family inet address 10.0.0.2/30
user@P1# set fe-1/2/0 unit 1 family mpls
user@P1# set fe-1/2/1 unit 4 description P1-to-P2
user@P1# set fe-1/2/1 unit 4 family inet address 10.0.0.29/30
user@P1# set fe-1/2/1 unit 4 family mpls
user@P1# set fe-1/2/2 unit 8 description P1-to-PE4
user@P1# set fe-1/2/2 unit 8 family inet address 10.0.0.17/30
user@P1# set fe-1/2/2 unit 8 family mpls
```

```
user@P1# set lo0 unit 3 family inet address 10.9.9.2/32
user@P1# set lo0 unit 3 family inet address 10.100.1.2/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@P1# set rsvp interface fe-1/2/0.1
user@P1# set rsvp interface fe-1/2/2.8
user@P1# set rsvp interface fe-1/2/1.4
user@P1# set mpls label-switched-path P1-to-P2 to 10.9.9.3
user@P1# set mpls label-switched-path P1-to-PE1 to 10.9.9.1
user@P1# set mpls label-switched-path P1-to-PE4 to 10.9.9.4
user@P1# set mpls interface fe-1/2/0.1
user@P1# set mpls interface fe-1/2/2.8
user@P1# set mpls interface fe-1/2/1.4
```

3. Configure BGP.

```
[edit protocols bgp group internal]
user@P1# set type internal
user@P1# set local-address 10.9.9.2
user@P1# set neighbor 10.9.9.1
user@P1# set neighbor 10.9.9.3
user@P1# set neighbor 10.9.9.4
```

4. Enable AIGP.

```
[edit protocols bgp group internal]
user@P1# set family inet labeled-unicast aigp
```

5. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf]
user@P1# set area 0.0.0.1 interface fe-1/2/0.1 metric 1
user@P1# set area 0.0.0.1 interface fe-1/2/1.4 metric 1
user@P1# set area 0.0.0.0 interface fe-1/2/2.8 metric 1
user@P1# set area 0.0.0.0 interface 10.9.9.2 passive
user@P1# set area 0.0.0.0 interface 10.9.9.2 metric 1
```

```
user@P1# set area 0.0.0.0 interface 10.100.1.2 passive
user@P1# set area 0.0.0.0 interface 10.100.1.2 metric 1
```

6. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@P1# set router-id 10.9.9.2
user@P1# set autonomous-system 13979
```

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

```
user@P1# commit
```

Results

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

```
user@P1# show interfaces
fe-1/2/0 {
    unit 1 {
        description P1-to-PE1;
        family inet {
            address 10.0.0.2/30;
        }
        family mpls;
    }
}
fe-1/2/1 {
    unit 4 {
        description P1-to-P2;
        family inet {
            address 10.0.0.29/30;
        }
        family mpls;
    }
}
fe-1/2/2 {
```

```
    unit 8 {
        description P1-to-PE4;
        family inet {
            address 10.0.0.17/30;
        }
        family mpls;
    }
}
lo0 {
    unit 3 {
        family inet {
            address 10.9.9.2/32;
            address 10.100.1.2/32;
        }
    }
}
```

```
user@P1# show protocols
rsvp {
    interface fe-1/2/0.1;
    interface fe-1/2/2.8;
    interface fe-1/2/1.4;
}
mpls {
    label-switched-path P1-to-P2 {
        to 10.9.9.3;
    }
    label-switched-path P1-to-PE1 {
        to 10.9.9.1;
    }
    label-switched-path P1-to-PE4 {
        to 10.9.9.4;
    }
    interface fe-1/2/0.1;
    interface fe-1/2/2.8;
    interface fe-1/2/1.4;
}
bgp {
    group internal {
        type internal;
        local-address 10.9.9.2;
```

```
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        neighbor 10.9.9.1;
        neighbor 10.9.9.3;
        neighbor 10.9.9.4;
    }
}
ospf {
    area 0.0.0.1 {
        interface fe-1/2/0.1 {
            metric 1;
        }
        interface fe-1/2/1.4 {
            metric 1;
        }
    }
    area 0.0.0.0 {
        interface fe-1/2/2.8 {
            metric 1;
        }
        interface 10.9.9.2 {
            passive;
            metric 1;
        }
        interface 10.100.1.2 {
            passive;
            metric 1;
        }
    }
}
```

```
user@P1# show routing-options
router-id 10.9.9.2;
autonomous-system 13979;
```

Configuring Device P2

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 Junos OS CLI User Guide.

To configure Device P2:

1. Configure the interfaces.

```
[edit interfaces]
user@P2# set fe-1/2/0 unit 3 description P2-to-PE1
user@P2# set fe-1/2/0 unit 3 family inet address 10.0.0.6/30
user@P2# set fe-1/2/0 unit 3 family mpls
user@P2# set fe-1/2/1 unit 5 description P2-to-P1
user@P2# set fe-1/2/1 unit 5 family inet address 10.0.0.30/30
user@P2# set fe-1/2/1 unit 5 family mpls
user@P2# set fe-1/2/2 unit 6 description P2-to-PE4
user@P2# set fe-1/2/2 unit 6 family inet address 10.0.0.13/30
user@P2# set fe-1/2/2 unit 6 family mpls
user@P2# set lo0 unit 5 family inet address 10.9.9.3/32
user@P2# set lo0 unit 5 family inet address 10.100.1.3/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@P2# set rsvp interface fe-1/2/1.5
user@P2# set rsvp interface fe-1/2/2.6
user@P2# set rsvp interface fe-1/2/0.3
user@P2# set mpls label-switched-path P2-to-PE1 to 10.9.9.1
user@P2# set mpls label-switched-path P2-to-P1 to 10.9.9.2
user@P2# set mpls label-switched-path P2-to-PE4 to 10.9.9.4
user@P2# set mpls interface fe-1/2/1.5
user@P2# set mpls interface fe-1/2/2.6
user@P2# set mpls interface fe-1/2/0.3
```

3. Configure BGP.

```
[edit protocols bgp group internal]
user@P2# set type internal
user@P2# set local-address 10.9.9.3
user@P2# set neighbor 10.9.9.1
user@P2# set neighbor 10.9.9.2
user@P2# set neighbor 10.9.9.4
```

4. Enable AIGP.

```
[edit protocols bgp group internal]
user@P2# set family inet labeled-unicast aigp
```

5. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf]
user@P2# set area 0.0.0.0 interface fe-1/2/2.6 metric 1
user@P2# set area 0.0.0.0 interface 10.9.9.3 passive
user@P2# set area 0.0.0.0 interface 10.9.9.3 metric 1
user@P2# set area 0.0.0.0 interface 10.100.1.3 passive
user@P2# set area 0.0.0.0 interface 10.100.1.3 metric 1
```

6. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@P2# set router-id 10.9.9.3
user@P2# set autonomous-system 13979
```

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

```
user@P2# commit
```

Results

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

```
user@P2# show interfaces
fe-1/2/0 {
    unit 3 {
        description P2-to-PE1;
        family inet {
            address 10.0.0.6/30;
        }
        family mpls;
    }
}
fe-1/2/1 {
    unit 5 {
        description P2-to-P1;
        family inet {
            address 10.0.0.30/30;
        }
        family mpls;
    }
}
fe-1/2/2 {
    unit 6 {
        description P2-to-PE4;
        family inet {
            address 10.0.0.13/30;
        }
        family mpls;
    }
}
lo0 {
    unit 5 {
        family inet {
            address 10.9.9.3/32;
            address 10.100.1.3/32;
        }
```

```
    }
}
```

```
user@P2# show protocols
rsvp {
    interface fe-1/2/1.5;
    interface fe-1/2/2.6;
    interface fe-1/2/0.3;
}
mpls {
    label-switched-path P2-to-PE1 {
        to 10.9.9.1;
    }
    label-switched-path P2-to-P1 {
        to 10.9.9.2;
    }
    label-switched-path P2-to-PE4 {
        to 10.9.9.4;
    }
    interface fe-1/2/1.5;
    interface fe-1/2/2.6;
    interface fe-1/2/0.3;
}
bgp {
    group internal {
        type internal;
        local-address 10.9.9.3;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        neighbor 10.9.9.1;
        neighbor 10.9.9.2;
        neighbor 10.9.9.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/2.6 {
            metric 1;
```

```
        }
        interface 10.9.9.3 {
            passive;
            metric 1;
        }
        interface 10.100.1.3 {
            passive;
            metric 1;
        }
    }
}
```

```
user@P2# show routing-options
router-id 10.9.9.3;
autonomous-system 13979;
```

Configuring Device PE4

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 Junos OS CLI User Guide.

To configure Device PE4:

1. Configure the interfaces.

```
[edit interfaces]
user@PE4# set fe-1/2/0 unit 7 description PE4-to-P2
user@PE4# set fe-1/2/0 unit 7 family inet address 10.0.0.14/30
user@PE4# set fe-1/2/0 unit 7 family mpls
user@PE4# set fe-1/2/1 unit 9 description PE4-to-P1
user@PE4# set fe-1/2/1 unit 9 family inet address 10.0.0.18/30
user@PE4# set fe-1/2/1 unit 9 family mpls
user@PE4# set fe-1/2/2 unit 10 description PE4-to-PE2
user@PE4# set fe-1/2/2 unit 10 family inet address 10.0.0.21/30
user@PE4# set fe-1/2/2 unit 10 family mpls
user@PE4# set fe-1/0/2 unit 12 description PE4-to-PE3
user@PE4# set fe-1/0/2 unit 12 family inet address 10.0.0.25/30
user@PE4# set fe-1/0/2 unit 12 family mpls
```

```
user@PE4# set lo0 unit 7 family inet address 10.9.9.4/32
user@PE4# set lo0 unit 7 family inet address 10.100.1.4/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@PE4# set rsvp interface fe-1/2/0.7
user@PE4# set rsvp interface fe-1/2/1.9
user@PE4# set rsvp interface fe-1/2/2.10
user@PE4# set rsvp interface fe-1/0/2.12
user@PE4# set mpls label-switched-path PE4-to-PE2 to 10.9.9.5
user@PE4# set mpls label-switched-path PE4-to-PE3 to 10.9.9.6
user@PE4# set mpls label-switched-path PE4-to-P1 to 10.9.9.2
user@PE4# set mpls label-switched-path PE4-to-P2 to 10.9.9.3
user@PE4# set mpls interface fe-1/2/0.7
user@PE4# set mpls interface fe-1/2/1.9
user@PE4# set mpls interface fe-1/2/2.10
user@PE4# set mpls interface fe-1/0/2.12
```

3. Configure BGP.

```
[edit protocols bgp]
user@PE4# set export next-hop
user@PE4# set export aigp
user@PE4# set group internal type internal
user@PE4# set group internal local-address 10.9.9.4
user@PE4# set group internal neighbor 10.9.9.1
user@PE4# set group internal neighbor 10.9.9.3
user@PE4# set group internal neighbor 10.9.9.2
user@PE4# set group external type external
user@PE4# set group external multihop ttl 2
user@PE4# set group external local-address 10.9.9.4
user@PE4# set group external peer-as 7018
user@PE4# set group external neighbor 10.9.9.5
user@PE4# set group external neighbor 10.9.9.6
```

4. Enable AIGP.

```
[edit protocols bgp]
user@PE4# set group external family inet labeled-unicast aigp
user@PE4# set group internal family inet labeled-unicast aigp
```

5. Originate a prefix, and configure an AIGP distance.

   By default, a prefix is originated using the current IGP distance. Optionally, you can configure a distance for the AIGP attribute, using the `distance` option, as shown here.

```
[edit policy-options policy-statement aigp term 10]
user@PE4# set from protocol static
user@PE4# set from route-filter 44.0.0.0/24 exact
user@PE4# set then aigp-originate distance 200
user@PE4# set then next-hop 10.100.1.4
user@PE4# set then accept
```

6. Enable the policies.

```
[edit policy-options policy-statement next-hop]
user@PE4# set term 10 from protocol bgp
user@PE4# set term 10 then next-hop 10.100.1.4
user@PE4# set term 10 then accept
user@PE4# set term 20 from protocol direct
user@PE4# set term 20 from route-filter 10.9.9.4/32 exact
user@PE4# set term 20 from route-filter 10.100.1.4/32 exact
user@PE4# set term 20 then next-hop 10.100.1.4
user@PE4# set term 20 then accept
```

7. Configure a static route.

```
[edit routing-options]
user@PE4# set static route 44.0.0.0/24 discard
```

8. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf]
user@PE4# set area 0.0.0.0 interface fe-1/2/1.9 metric 1
user@PE4# set area 0.0.0.0 interface fe-1/2/0.7 metric 1
user@PE4# set area 0.0.0.0 interface 10.9.9.4 passive
user@PE4# set area 0.0.0.0 interface 10.9.9.4 metric 1
user@PE4# set area 0.0.0.0 interface 10.100.1.4 passive
user@PE4# set area 0.0.0.0 interface 10.100.1.4 metric 1
user@PE4# set area 0.0.0.2 interface fe-1/2/2.10 metric 1
user@PE4# set area 0.0.0.3 interface fe-1/0/2.12 metric 1
```

9. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@PE4# set router-id 10.9.9.4
user@PE4# set autonomous-system 13979
```

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

```
user@PE4# commit
```

Results

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

```
user@PE4# show interfaces
fe-1/0/2 {
    unit 12 {
        description PE4-to-PE3;
        family inet {
            address 10.0.0.25/30;
        }
        family mpls;
    }
}
```

```
fe-1/2/0 {
    unit 7 {
        description PE4-to-P2;
        family inet {
            address 10.0.0.14/30;
        }
        family mpls;
    }
}
fe-1/2/1 {
    unit 9 {
        description PE4-to-P1;
        family inet {
            address 10.0.0.18/30;
        }
        family mpls;
    }
}
fe-1/2/2 {
    unit 10 {
        description PE4-to-PE2;
        family inet {
            address 10.0.0.21/30;
        }
        family mpls;
    }
}
lo0 {
    unit 7 {
        family inet {
            address 10.9.9.4/32;
            address 10.100.1.4/32;
        }
    }
}
```

```
user@PE4# show policy-options
policy-statement aigp {
    term 10 {
        from {
            protocol static;
```

```
            route-filter 44.0.0.0/24 exact;
        }
        then {
            aigp-originate distance 200;
            next-hop 10.100.1.4;
            accept;
        }
    }
}
policy-statement next-hop {
    term 10 {
        from protocol bgp;
        then {
            next-hop 10.100.1.4;
            accept;
        }
    }
    term 20 {
        from {
            protocol direct;
            route-filter 10.9.9.4/32 exact;
            route-filter 10.100.1.4/32 exact;
        }
        then {
            next-hop 10.100.1.4;
            accept;
        }
    }
}
```

```
user@PE4# show protocols
rsvp {
    interface fe-1/2/0.7;
    interface fe-1/2/1.9;
    interface fe-1/2/2.10;
    interface fe-1/0/2.12;
}
mpls {
    label-switched-path PE4-to-PE2 {
        to 10.9.9.5;
    }
```

```
    label-switched-path PE4-to-PE3 {
        to 10.9.9.6;
    }
    label-switched-path PE4-to-P1 {
        to 10.9.9.2;
    }
    label-switched-path PE4-to-P2 {
        to 10.9.9.3;
    }
    interface fe-1/2/0.7;
    interface fe-1/2/1.9;
    interface fe-1/2/2.10;
    interface fe-1/0/2.12;
}
bgp {
    export [ next-hop aigp ];
    group internal {
        type internal;
        local-address 10.9.9.4;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        neighbor 10.9.9.1;
        neighbor 10.9.9.3;
        neighbor 10.9.9.2;
    }
    group external {
        type external;
        multihop {
            ttl 2;
        }
        local-address 10.9.9.4;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        peer-as 7018;
        neighbor 10.9.9.5;
        neighbor 10.9.9.6;
    }
```

```
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.9 {
            metric 1;
        }
        interface fe-1/2/0.7 {
            metric 1;
        }
        interface 10.9.9.4 {
            passive;
            metric 1;
        }
        interface 10.100.1.4 {
            passive;
            metric 1;
        }
    }
    area 0.0.0.2 {
        interface fe-1/2/2.10 {
            metric 1;
        }
    }
    area 0.0.0.3 {
        interface fe-1/0/2.12 {
            metric 1;
        }
    }
}
```

```
user@PE4# show routing-options
static {
    route 44.0.0.0/24 discard;
}
router-id 10.9.9.4;
autonomous-system 13979;
```

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 Junos OS CLI User Guide.

To configure Device PE1:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set fe-1/2/0 unit 0 description PE1-to-P1
user@PE1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30
user@PE1# set fe-1/2/0 unit 0 family mpls
user@PE1# set fe-1/2/1 unit 2 description PE1-to-P2
user@PE1# set fe-1/2/1 unit 2 family inet address 10.0.0.5/30
user@PE1# set fe-1/2/1 unit 2 family mpls
user@PE1# set fe-1/2/2 unit 14 description PE1-to-PE7
user@PE1# set fe-1/2/2 unit 14 family inet address 10.0.0.9/30
user@PE1# set lo0 unit 1 family inet address 10.9.9.1/32
user@PE1# set lo0 unit 1 family inet address 10.100.1.1/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@PE1# set rsvp interface fe-1/2/0.0
user@PE1# set rsvp interface fe-1/2/1.2
user@PE1# set rsvp interface fe-1/2/2.14
user@PE1# set mpls label-switched-path PE1-to-P1 to 10.9.9.2
user@PE1# set mpls label-switched-path PE1-to-P2 to 10.9.9.3
user@PE1# set mpls interface fe-1/2/0.0
user@PE1# set mpls interface fe-1/2/1.2
user@PE1# set mpls interface fe-1/2/2.14
```

3. Configure BGP.

```
[edit protocols bgp]
user@PE1# set group internal type internal
user@PE1# set group internal local-address 10.9.9.1
```

```
user@PE1# set group internal export SET_EXPORT_ROUTES
user@PE1# set group internal vpn-apply-export
user@PE1# set group internal neighbor 10.9.9.4
user@PE1# set group internal neighbor 10.9.9.2
user@PE1# set group internal neighbor 10.9.9.3
user@PE1# set group external type external
user@PE1# set group external export SET_EXPORT_ROUTES
user@PE1# set group external peer-as 7019
user@PE1# set group external neighbor 10.0.0.10
```

4. Enable AIGP.

```
[edit protocols bgp]
user@PE1# set group internal family inet labeled-unicast aigp
user@PE1# set group external family inet labeled-unicast aigp
```

5. Enable the policies.

```
[edit policy-options policy-statement SET_EXPORT_ROUTES term 10]
user@PE1# set from protocol direct
user@PE1# set from protocol bgp
user@PE1# set then next-hop 10.100.1.1
user@PE1# set then accept
```

6. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf area 0.0.0.1]
user@PE1# set interface fe-1/2/0.0 metric 1
user@PE1# set interface fe-1/2/1.2 metric 1
user@PE1# set interface 10.9.9.1 passive
user@PE1# set interface 10.9.9.1 metric 1
user@PE1# set interface 10.100.1.1 passive
user@PE1# set interface 10.100.1.1 metric 1
```

7. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@PE1# set router-id 10.9.9.1
user@PE1# set autonomous-system 13979
```

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

```
user@PE1# commit
```

Results

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

```
user@PE1# show interfaces
fe-1/2/0 {
    unit 0 {
        description PE1-to-P1;
        family inet {
            address 10.0.0.1/30;
        }
        family mpls;
    }
}
fe-1/2/1 {
    unit 2 {
        description PE1-to-P2;
        family inet {
            address 10.0.0.5/30;
        }
        family mpls;
    }
}
fe-1/2/2 {
    unit 14 {
        description PE1-to-PE7;
        family inet {
```

```
            address 10.0.0.9/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.9.9.1/32;
            address 10.100.1.1/32;
        }
    }
}
```

```
user@PE1# show policy-options
policy-statement SET_EXPORT_ROUTES {
    term 10 {
        from protocol [ direct bgp ];
        then {
            next-hop 10.100.1.1;
            accept;
        }
    }
}
```

```
user@PE1# show protocols
rsvp {
    interface fe-1/2/0.0;
    interface fe-1/2/1.2;
    interface fe-1/2/2.14;
}
mpls {
    label-switched-path PE1-to-P1 {
        to 10.9.9.2;
    }
    label-switched-path PE1-to-P2 {
        to 10.9.9.3;
    }
    interface fe-1/2/0.0;
    interface fe-1/2/1.2;
    interface fe-1/2/2.14;
```

```
}
bgp {
    group internal {
        type internal;
        local-address 10.9.9.1;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        export SET_EXPORT_ROUTES;
        vpn-apply-export;
        neighbor 10.9.9.4;
        neighbor 10.9.9.2;
        neighbor 10.9.9.3;
    }
    group external {
        type external;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        export SET_EXPORT_ROUTES;
        peer-as 7019;
        neighbor 10.0.0.10;
    }
}
ospf {
    area 0.0.0.1 {
        interface fe-1/2/0.0 {
            metric 1;
        }
        interface fe-1/2/1.2 {
            metric 1;
        }
        interface 10.9.9.1 {
            passive;
            metric 1;
        }
        interface 10.100.1.1 {
            passive;
            metric 1;
```

```
        }
    }
}
```

```
user@PE1# show routing-options
router-id 10.9.9.1;
autonomous-system 13979;
```

Configuring Device PE2

Step-by-Step Procedure

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

To configure Device PE2:

1. Configure the interfaces.

```
[edit interfaces]
user@PE2# set fe-1/2/0 unit 11 description PE2-to-PE4
user@PE2# set fe-1/2/0 unit 11 family inet address 10.0.0.22/30
user@PE2# set fe-1/2/0 unit 11 family mpls
user@PE2# set lo0 unit 9 family inet address 10.9.9.5/32 primary
user@PE2# set lo0 unit 9 family inet address 10.100.1.5/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@PE2# set rsvp interface fe-1/2/0.11
user@PE2# set mpls label-switched-path PE2-to-PE4 to 10.9.9.4
user@PE2# set mpls interface fe-1/2/0.11
```

3. Configure BGP.

```
[edit protocols bgp]
user@PE2# set group external type external
user@PE2# set group external multihop ttl 2
```

```
user@PE2# set group external local-address 10.9.9.5
user@PE2# set group external export next-hop
user@PE2# set group external export aigp
user@PE2# set group external export SET_EXPORT_ROUTES
user@PE2# set group external vpn-apply-export
user@PE2# set group external peer-as 13979
user@PE2# set group external neighbor 10.9.9.4
```

4. Enable AIGP.

```
[edit protocols bgp]
user@PE2# set group external family inet labeled-unicast aigp
```

5. Originate a prefix, and configure an AIGP distance.

   By default, a prefix is originated using the current IGP distance. Optionally, you can configure a distance for the AIGP attribute, using the `distance` option, as shown here.

```
[edit policy-options policy-statement aigp]
user@PE2# set term 10 from route-filter 55.0.0.0/24 exact
user@PE2# set term 10 then aigp-originate distance 20
user@PE2# set term 10 then next-hop 10.100.1.5
user@PE2# set term 10 then accept
user@PE2# set term 20 from route-filter 99.0.0.0/24 exact
user@PE2# set term 20 then aigp-originate distance 30
user@PE2# set term 20 then next-hop 10.100.1.5
user@PE2# set term 20 then accept
```

6. Enable the policies.

```
[edit policy-options]
user@PE2# set policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
user@PE2# set policy-statement SET_EXPORT_ROUTES term 10 from protocol static
user@PE2# set policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
user@PE2# set policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.5
user@PE2# set policy-statement SET_EXPORT_ROUTES term 10 then accept
user@PE2# set policy-statement next-hop term 10 from protocol bgp
user@PE2# set policy-statement next-hop term 10 then next-hop 10.100.1.5
user@PE2# set policy-statement next-hop term 10 then accept
user@PE2# set policy-statement next-hop term 20 from protocol direct
```

```
user@PE2# set policy-statement next-hop term 20 from route-filter 10.9.9.5/32 exact
user@PE2# set policy-statement next-hop term 20 from route-filter 10.100.1.5/32 exact
user@PE2# set policy-statement next-hop term 20 then next-hop 10.100.1.5
user@PE2# set policy-statement next-hop term 20 then accept
```

7. Enable some static routes.

```
[edit routing-options]
user@PE2# set static route 99.0.0.0/24 discard
user@PE2# set static route 55.0.0.0/24 discard
```

8. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf area 0.0.0.2]
user@PE2# set interface 10.9.9.5 passive
user@PE2# set interface 10.9.9.5 metric 1
user@PE2# set interface 10.100.1.5 passive
user@PE2# set interface 10.100.1.5 metric 1
user@PE2# set interface fe-1/2/0.11 metric 1
```

9. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@PE2# set router-id 10.9.9.5
user@PE2# set autonomous-system 7018
```

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

```
user@PE2# commit
```

Results

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

```
user@PE2# show interfaces
fe-1/2/0 {
    unit 11 {
        description PE2-to-PE4;
        family inet {
            address 10.0.0.22/30;
        }
        family mpls;
    }
}
lo0 {
    unit 9 {
        family inet {
            address 10.9.9.5/32 {
                primary;
            }
            address 10.100.1.5/32;
        }
    }
}
```

```
user@PE2# show policy-options
policy-statement SET_EXPORT_ROUTES {
    term 10 {
        from protocol [ direct static bgp ];
        then {
            next-hop 10.100.1.5;
            accept;
        }
    }
}
policy-statement aigp {
    term 10 {
        from {
            route-filter 55.0.0.0/24 exact;
```

```
        }
        then {
            aigp-originate distance 20;
            next-hop 10.100.1.5;
            accept;
        }
    }
    term 20 {
        from {
            route-filter 99.0.0.0/24 exact;
        }
        then {
            aigp-originate distance 30;
            next-hop 10.100.1.5;
            accept;
        }
    }
}
policy-statement next-hop {
    term 10 {
        from protocol bgp;
        then {
            next-hop 10.100.1.5;
            accept;
        }
    }
    term 20 {
        from {
            protocol direct;
            route-filter 10.9.9.5/32 exact;
            route-filter 10.100.1.5/32 exact;
        }
        then {
            next-hop 10.100.1.5;
            accept;
        }
    }
}
```

```
user@PE2# show protocols
rsvp {
```

```
    interface fe-1/2/0.11;
}
mpls {
    label-switched-path PE2-to-PE4 {
        to 10.9.9.4;
    }
    interface fe-1/2/0.11;
}
bgp {
    group external {
        type external;
        multihop {
            ttl 2;
        }
        local-address 10.9.9.5;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        export [ next-hop aigp SET_EXPORT_ROUTES ];
        vpn-apply-export;
        peer-as 13979;
        neighbor 10.9.9.4;
    }
}
ospf {
    area 0.0.0.2 {
        interface 10.9.9.5 {
            passive;
            metric 1;
        }
        interface 10.100.1.5 {
            passive;
            metric 1;
        }
        interface fe-1/2/0.11 {
            metric 1;
        }
```

```
    }
}
```

```
user@PE2# show routing-options
static {
    route 99.0.0.0/24 discard;
    route 55.0.0.0/24 discard;
}
router-id 10.9.9.5;
autonomous-system 7018;
```

Configuring Device PE3

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 Junos OS CLI User Guide.

To configure Device PE3:

1. Configure the interfaces.

```
[edit interfaces]
user@PE3# set fe-1/2/0 unit 13 description PE3-to-PE4
user@PE3# set fe-1/2/0 unit 13 family inet address 10.0.0.26/30
user@PE3# set fe-1/2/0 unit 13 family mpls
user@PE3# set lo0 unit 11 family inet address 10.9.9.6/32
user@PE3# set lo0 unit 11 family inet address 10.100.1.6/32
```

2. Configure MPLS and a signaling protocol, such as RSVP or LDP.

```
[edit protocols]
user@PE3# set rsvp interface fe-1/2/0.13
user@PE3# set mpls label-switched-path PE3-to-PE4 to 10.9.9.4
user@PE3# set mpls interface fe-1/2/0.13
```

3. Configure BGP.

```
[edit protocols bgp group external]
user@PE3# set type external
user@PE3# set multihop ttl 2
user@PE3# set local-address 10.9.9.6
user@PE3# set export next-hop
user@PE3# set export SET_EXPORT_ROUTES
user@PE3# set vpn-apply-export
user@PE3# set peer-as 13979
user@PE3# set neighbor 10.9.9.4
```

4. Enable AIGP.

```
[edit protocols bgp group external]
user@PE3# set family inet labeled-unicast aigp
```

5. Enable the policies.

```
[edit policy-options]
user@PE3# set policy-statement SET_EXPORT_ROUTES term 10 from protocol direct
user@PE3# set policy-statement SET_EXPORT_ROUTES term 10 from protocol static
user@PE3# set policy-statement SET_EXPORT_ROUTES term 10 from protocol bgp
user@PE3# set policy-statement SET_EXPORT_ROUTES term 10 then next-hop 10.100.1.6
user@PE3# set policy-statement SET_EXPORT_ROUTES term 10 then accept
user@PE3# set policy-statement next-hop term 10 from protocol bgp
user@PE3# set policy-statement next-hop term 10 then next-hop 10.100.1.6
user@PE3# set policy-statement next-hop term 10 then accept
user@PE3# set policy-statement next-hop term 20 from protocol direct
user@PE3# set policy-statement next-hop term 20 from route-filter 10.9.9.6/32 exact
user@PE3# set policy-statement next-hop term 20 from route-filter 10.100.1.6/32 exact
user@PE3# set policy-statement next-hop term 20 then next-hop 10.100.1.6
user@PE3# set policy-statement next-hop term 20 then accept
```

6. Configure an IGP, such as OSPF, RIP, or IS-IS.

```
[edit protocols ospf area 0.0.0.3]
user@PE3# set interface 10.9.9.6 passive
user@PE3# set interface 10.9.9.6 metric 1
```

```
user@PE3# set interface 10.100.1.6 passive
user@PE3# set interface 10.100.1.6 metric 1
user@PE3# set interface fe-1/2/0.13 metric 1
```

7. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@PE3# set router-id 10.9.9.6
user@PE3# set autonomous-system 7018
```

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

```
user@PE3# commit
```

Results

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

```
user@PE3# show interfaces
fe-1/2/0 {
    unit 13 {
        description PE3-to-PE4;
        family inet {
            address 10.0.0.26/30;
        }
        family mpls;
    }
}
lo0 {
    unit 11 {
        family inet {
            address 10.9.9.6/32;
            address 10.100.1.6/32;
        }
```

```
    }
}
```

```
user@PE3# show policy-options
policy-statement SET_EXPORT_ROUTES {
    term 10 {
        from protocol [ direct static bgp ];
        then {
            next-hop 10.100.1.6;
            accept;
        }
    }
}
policy-statement next-hop {
    term 10 {
        from protocol bgp;
        then {
            next-hop 10.100.1.6;
            accept;
        }
    }
    term 20 {
        from {
            protocol direct;
            route-filter 10.9.9.6/32 exact;
            route-filter 10.100.1.6/32 exact;
        }
        then {
            next-hop 10.100.1.6;
            accept;
        }
    }
}
```

```
user@PE3# show protocols
rsvp {
    interface fe-1/2/0.13;
}
mpls {
    label-switched-path PE3-to-PE4 {
```

```
        to 10.9.9.4;
    }
    interface fe-1/2/0.13;
}
bgp {
    group external {
        type external;
        multihop {
            ttl 2;
        }
        local-address 10.9.9.6;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        export [ next-hop SET_EXPORT_ROUTES ];
        vpn-apply-export;
        peer-as 13979;
        neighbor 10.9.9.4;
    }
}
ospf {
    area 0.0.0.3 {
        interface 10.9.9.6 {
            passive;
            metric 1;
        }
        interface 10.100.1.6 {
            passive;
            metric 1;
        }
        interface fe-1/2/0.13 {
            metric 1;
        }
    }
}
```

```
user@PE3# show routing-options
router-id 10.9.9.6;
autonomous-system 7018;
```

Configuring Device PE7

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 Junos OS CLI User Guide.

To configure Device PE7:

1. Configure the interfaces.

```
[edit interfaces]
user@PE7# set fe-1/2/0 unit 15 description PE7-to-PE1
user@PE7# set fe-1/2/0 unit 15 family inet address 10.0.0.10/30
user@PE7# set lo0 unit 13 family inet address 10.9.9.7/32
user@PE7# set lo0 unit 13 family inet address 10.100.1.7/32
```

2. Configure BGP.

```
[edit protocols bgp group external]
user@PE7# set type external
user@PE7# set export SET_EXPORT_ROUTES
user@PE7# set peer-as 13979
user@PE7# set neighbor 10.0.0.9
```

3. Enable AIGP.

```
[edit protocols bgp group external]
user@PE7# set family inet labeled-unicast aigp
```

4. Configure the routing policy.

```
[edit policy-options policy-statement SET_EXPORT_ROUTES term 10]
user@PE7# set from protocol direct
user@PE7# set from protocol bgp
user@PE7# set then next-hop 10.100.1.7
user@PE7# set then accept
```

5. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@PE7# set router-id 10.9.9.7
user@PE7# set autonomous-system 7019
```

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

```
user@PE7# commit
```

Results

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

```
user@PE7# show interfaces
interfaces {
    fe-1/2/0 {
        unit 15 {
            description PE7-to-PE1;
            family inet {
                address 10.0.0.10/30;
            }
        }
    }
    lo0 {
        unit 13 {
            family inet {
                address 10.9.9.7/32;
                address 10.100.1.7/32;
            }
        }
    }
}
```

```
user@PE7# show policy-options
policy-statement SET_EXPORT_ROUTES {
```

```
    term 10 {
        from protocol [ direct bgp ];
        then {
            next-hop 10.100.1.7;
            accept;
        }
    }
}
```

```
user@PE7# show protocols
bgp {
    group external {
        type external;
        family inet {
            labeled-unicast {
                aigp;
            }
        }
        export SET_EXPORT_ROUTES;
        peer-as 13979;
        neighbor 10.0.0.9;
    }
}
```

```
user@PE7# show routing-options
router-id 10.9.9.7;
autonomous-system 7019;
```

Verification

IN THIS SECTION

- Verifying That Device PE4 Is Receiving the AIGP Attribute from Its EBGP Neighbor PE2 | **218**
- Checking the IGP Metric | **219**
- Verifying That Device PE4 Adds the IGP Metric to the AIGP Attribute | **219**
- Verifying That Device PE7 Is Receiving the AIGP Attribute from Its EBGP Neighbor PE1 | **220**
- Verifying the Resolving AIGP Metric | **221**

- Verifying the Presence of AIGP Attributes in BGP Updates | 225

Confirm that the configuration is working properly.

Verifying That Device PE4 Is Receiving the AIGP Attribute from Its EBGP Neighbor PE2

Purpose

Make sure that the AIGP policy on Device PE2 is working.

Action

```
user@PE4> show route receive-protocol bgp 10.9.9.5 extensive
* 55.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 299888
     Nexthop: 10.100.1.5
     AS path: 7018 I
     AIGP: 20

* 99.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 299888
     Nexthop: 10.100.1.5
     AS path: 7018 I
     AIGP: 30
```

Meaning

On Device PE2, the `aigp-originate` statement is configured with a distance of 20 (`aigp-originate distance 20`). This statement is applied to route 55.0.0.0/24. Likewise, the `aigp-originate distance 30` statement is applied to route 99.0.0.0/24. Thus, when Device PE4 receives these routes, the AIGP attribute is attached with the configured metrics.

Checking the IGP Metric

Purpose

From Device PE4, check the IGP metric to the BGP next hop 10.100.1.5.

Action

```
user@PE4> show route 10.100.1.5
inet.0: 30 destinations, 40 routes (30 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.100.1.5/32      *[OSPF/10] 05:35:50, metric 2
                    > to 10.0.0.22 via fe-1/2/2.10
                    [BGP/170] 03:45:07, localpref 100, from 10.9.9.5
                      AS path: 7018 I
                    > to 10.0.0.22 via fe-1/2/2.10
```

Meaning

The IGP metric for this route is 2.

Verifying That Device PE4 Adds the IGP Metric to the AIGP Attribute

Purpose

Make sure that Device PE4 adds the IGP metric to the AIGP attribute when it readvertises routes to its IBGP neighbor, Device PE1.

Action

```
user@PE4> show route advertising-protocol bgp 10.9.9.1 extensive

* 55.0.0.0/24 (1 entry, 1 announced)
 BGP group internal type Internal
     Route Label: 300544
     Nexthop: 10.100.1.4
     Flags: Nexthop Change
     Localpref: 100
     AS path: [13979] 7018 I
```

```
     AIGP: 22

* 99.0.0.0/24 (1 entry, 1 announced)
 BGP group internal type Internal
     Route Label: 300544
     Nexthop: 10.100.1.4
     Flags: Nexthop Change
     Localpref: 100
     AS path: [13979] 7018 I
     AIGP: 32
```

Meaning

The IGP metric is added to the AIGP metric (20 + 2 = 22 and 30 + 2 = 32), because the next hop is changed for these routes.

Verifying That Device PE7 Is Receiving the AIGP Attribute from Its EBGP Neighbor PE1

Purpose

Make sure that the AIGP policy on Device PE1 is working.

Action

```
user@PE7> show route receive-protocol bgp 10.0.0.9 extensive
* 44.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300096
     Nexthop: 10.0.0.9
     AS path: 13979 I
     AIGP: 203

* 55.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300112
     Nexthop: 10.0.0.9
     AS path: 13979 7018 I
     AIGP: 25

* 99.0.0.0/24 (1 entry, 1 announced)
     Accepted
```

```
    Route Label: 300112
    Nexthop: 10.0.0.9
    AS path: 13979 7018 I
    AIGP: 35
```

Meaning

The 44.0.0.0/24 route is originated at Device PE4. The 55.0.0.0/24 and 99.0.0.0/24 routes are originated at Device PE2. The IGP distances are added to the configured AIGP distances.

Verifying the Resolving AIGP Metric

Purpose

Confirm that if the prefix is resolved through recursion and the recursive next hops have AIGP metrics, the prefix has the sum of the AIGP values that are on the recursive BGP next hops.

Action

1. Add a static route to 66.0.0.0/24.

```
[edit routing-options]
user@PE2# set static route 66.0.0.0/24 discard
```

2. Delete the existing terms in the `aigp` policy statement on Device PE2.

```
[edit policy-options policy-statement aigp]
user@PE2# delete term 10
user@PE2# delete term 20
```

3. Configure a recursive route lookup for the route to 66.0.0.0.

   The policy shows the AIGP metric for prefix 66.0.0.0/24 (none) and its recursive next hop. Prefix 66.0.0.0/24 is resolved by 55.0.0.1. Prefix 66.0.0.0/24 does not have its own AIGP metric being originated, but its recursive next hop, 55.0.0.1, has an AIGP value.

```
[edit policy-options policy-statement aigp]
user@PE2# set term 10 from route-filter 55.0.0.1/24 exact
user@PE2# set term 10 then aigp-originate distance 20
```

```
user@PE2# set term 10 then next-hop 10.100.1.5
user@PE2# set term 10 then accept
user@PE2# set term 20 from route-filter 66.0.0.0/24 exact
user@PE2# set term 20 then next-hop 55.0.0.1
user@PE2# set term 20 then accept
```

4. On Device PE4, run the `show route 55.0.0.0 extensive` command.

   The value of Metric2 is the IGP metric to the BGP next hop. When Device PE4 readvertises these routes to its IBGP peer, Device PE1, the AIGP metric is the sum of AIGP + its Resolving AIGP metric + Metric2.

   Prefix 55.0.0.0 shows its own IGP metric 20, as defined and advertised by Device PE2. It does not show a resolving AIGP value because it does not have a recursive BGP next hop. The value of Metric2 is 2.

```
user@PE4> show route 55.0.0.0 extensive
inet.0: 31 destinations, 41 routes (31 active, 0 holddown, 0 hidden)
55.0.0.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 55.0.0.0/24 -> {indirect(262151)}
Page 0 idx 0 Type 1 val 928d1b8
    Flags: Nexthop Change
    Nexthop: 10.100.1.4
    Localpref: 100
    AS path: [13979] 7018 I
    Communities:
    AIGP: 22
Path 55.0.0.0 from 10.9.9.5 Vector len 4.  Val: 0
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x925da38
                Next-hop reference count: 4
                Source: 10.9.9.5
                Next hop type: Router, Next hop index: 1004
                Next hop: 10.0.0.22 via fe-1/2/2.10, selected
                Label operation: Push 299888
                Label TTL action: prop-ttl
                Protocol next hop: 10.100.1.5
                Push 299888
                Indirect next hop: 93514d8 262151
                State: <Active Ext>
```

```
                Local AS: 13979 Peer AS:  7018
                Age: 22:03:26   Metric2: 2
                AIGP: 20
                Task: BGP_7018.10.9.9.5+58560
                Announcement bits (3): 3-KRT 4-BGP_RT_Background 5-Resolve tree 1
                AS path: 7018 I
                Accepted
                Route Label: 299888
                Localpref: 100
                Router ID: 10.9.9.5
                Indirect next hops: 1
                        Protocol next hop: 10.100.1.5 Metric: 2
                        Push 299888
                        Indirect next hop: 93514d8 262151
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.0.0.22 via fe-1/2/2.10
                        10.100.1.5/32 Originating RIB: inet.0
                          Metric: 2                     Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.0.0.22 via fe-1/2/2.10
```

5. On Device PE4, run the `show route 66.0.0.0 extensive` command.

   Prefix 66.0.0.0/24 shows the Resolving AIGP, which is the sum of its own AIGP metric and its recursive BGP next hop:

   66.0.0.1 = 0, 55.0.0.1 = 20, 0+20 = 20

```
user@PE4> show route 66.0.0.0 extensive
inet.0: 31 destinations, 41 routes (31 active, 0 holddown, 0 hidden)
66.0.0.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 66.0.0.0/24 -> {indirect(262162)}
Page 0 idx 0 Type 1 val 928cefc
    Flags: Nexthop Change
    Nexthop: 10.100.1.4
    Localpref: 100
    AS path: [13979] 7018 I
    Communities:
Path 66.0.0.0 from 10.9.9.5 Vector len 4.  Val: 0
        *BGP    Preference: 170/-101
```

```
                Next hop type: Indirect
                Address: 0x925d4e0
                Next-hop reference count: 4
                Source: 10.9.9.5
                Next hop type: Router, Next hop index: 1006
                Next hop: 10.0.0.22 via fe-1/2/2.10, selected
                Label operation: Push 299888, Push 299888(top)
                Label TTL action: prop-ttl, prop-ttl(top)
                Protocol next hop: 55.0.0.1
                Push 299888
                Indirect next hop: 9353e88 262162
                State: <Active Ext>
                Local AS: 13979 Peer AS:  7018
                Age: 31:42      Metric2: 2
                Resolving-AIGP: 20
                Task: BGP_7018.10.9.9.5+58560
                Announcement bits (3): 3-KRT 4-BGP_RT_Background 5-Resolve tree 1
                AS path: 7018 I
                Accepted
                Route Label: 299888
                Localpref: 100
                Router ID: 10.9.9.5
                Indirect next hops: 1
                        Protocol next hop: 55.0.0.1 Metric: 2 AIGP: 20
                        Push 299888
                        Indirect next hop: 9353e88 262162
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.0.0.22 via fe-1/2/2.10
                        55.0.0.0/24 Originating RIB: inet.0
                          Metric: 2                     Node path count: 1
                          Indirect nexthops: 1
                                Protocol Nexthop: 10.100.1.5 Metric: 2 Push 299888
                                Indirect nexthop: 93514d8 262151
                                Indirect path forwarding nexthops: 1
                                        Nexthop: 10.0.0.22 via fe-1/2/2.10
                                10.100.1.5/32 Originating RIB: inet.0
                                  Metric: 2                             Node path count: 1
                                  Forwarding nexthops: 1
                                        Nexthop: 10.0.0.22 via fe-1/2/2.10
```

Verifying the Presence of AIGP Attributes in BGP Updates

Purpose

If the AIGP attribute is not enabled under BGP (or the `group` or `neighbor` hierarchies), the AIGP attribute is silently discarded. Enable **traceoptions** and include the **packets** flag in the **detail** option in the configuration to confirm the presence of the AIGP attribute in transmitted or received BGP updates. This is useful when debugging AIGP issues.

Action

1. Configure Device PE2 and Device PE4 for **traceoptions**.

```
user@host> show protocols bgp
    traceoptions {
        file bgp size 1m files 5;
        flag packets detail;
    }
```

2. Check the **traceoptions** file on Device PE2.

   The following sample shows Device PE2 advertising prefix 99.0.0.0/24 to Device PE4 (10.9.9.4) with an AIGP metric of 20:

```
user@PE2> show log bgp
Mar 22 09:27:18.982150 BGP SEND 10.9.9.5+49652 -> 10.9.9.4+179
Mar 22 09:27:18.982178 BGP SEND message type 2 (Update) length 70
Mar 22 09:27:18.982198 BGP SEND Update PDU length 70
Mar 22 09:27:18.982248 BGP SEND flags 0x40 code Origin(1): IGP
Mar 22 09:27:18.982273 BGP SEND flags 0x40 code ASPath(2) length 6: 7018
Mar 22 09:27:18.982295 BGP SEND flags 0x80 code AIGP(26): AIGP: 20
Mar 22 09:27:18.982316 BGP SEND flags 0x90 code MP_reach(14): AFI/SAFI 1/4
Mar 22 09:27:18.982341 BGP SEND         nhop 10.100.1.5 len 4
Mar 22 09:27:18.982372 BGP SEND         99.0.0.0/24 (label 301664)
Mar 22 09:27:33.665412 bgp_send: sending 19 bytes to abcd::10:255:170:84 (External AS 13979)
```

3. Verify that the route was received on Device PE4 using the **show route receive-protocol** command.

AIGP is not enabled on Device PE4, so the AIGP attribute is silently discarded for prefix 99.0.0.0/24 and does not appear in the following output:

```
user@PE4> show route receive-protocol bgp 10.9.9.5 extensive | find 55.0.0.0
* 99.0.0.0/24 (2 entries, 1 announced)
     Accepted
     Route Label: 301728
     Nexthop: 10.100.1.5
     AS path: 7018 I
```

4. Check the **traceoptions** file on Device PE4.

   The following output from the **traceoptions** log shows that the 99.0.0.0/24 prefix was received with the AIGP attribute attached:

```
user@PE4> show log bgp
Mar 22 09:41:39.650295 BGP RECV 10.9.9.5+64690 -> 10.9.9.4+179
Mar 22 09:41:39.650331 BGP RECV message type 2 (Update) length 70
Mar 22 09:41:39.650350 BGP RECV Update PDU length 70
Mar 22 09:41:39.650370 BGP RECV flags 0x40 code Origin(1): IGP
Mar 22 09:41:39.650394 BGP RECV flags 0x40 code ASPath(2) length 6: 7018
Mar 22 09:41:39.650415 BGP RECV flags 0x80 code AIGP(26): AIGP: 20
Mar 22 09:41:39.650436 BGP RECV flags 0x90 code MP_reach(14): AFI/SAFI 1/4
Mar 22 09:41:39.650459 BGP RECV         nhop 10.100.1.5 len 4
Mar 22 09:41:39.650495 BGP RECV         99.0.0.0/24 (label 301728)
Mar 22 09:41:39.650574 bgp_rcv_nlri: 99.0.0.0/24
Mar 22 09:41:39.650607 bgp_rcv_nlri: 99.0.0.0/24 belongs to meshgroup
Mar 22 09:41:39.650629 bgp_rcv_nlri: 99.0.0.0/24 qualified bnp->ribact 0x0 l2afcb 0x0
```

Meaning

Performing this verification helps with AIGP troubleshooting and debugging issues. It enables you to verify which devices in your network send and receive AIGP attributes.

SEE ALSO

Example: Enabling BGP Route Advertisements | 242

Understanding AS Override

The AS override feature allows a provider edge (PE) router to change the private autonomous system (AS) number used by a customer edge (CE) device on an external BGP (EBGP) session running on a VPN routing and forwarding (VRF) access link. The private AS number is changed to the PE AS number. Another CE device connected to another PE device sees the EBGP route coming from the first site with an AS path of provider-ASN provider-ASN, instead of provider-ASN site1-ASN. This allows enterprise networks to use the same private ASN on all sites.

The AS override feature offers a clear management advantage to the service provider because BGP by default does not accept BGP routes with an AS path attribute that contains the local AS number.

In an enterprise network with multiple sites, you might wish to use a single AS number across sites. Suppose, for example that two CE devices are in AS 64512 and that the provider network is in AS 65534.

When the service provider configures a Layer 3 VPN with this setup, even if the MPLS network has routes towards Device CE1 and Device CE2, Device CE1 and Device CE2 do not have routes to each other because the AS path attribute would appear as 64512 65534 64512. BGP uses the AS path attribute as its loop avoidance mechanism. If a site sees its own AS number more than once in the AS path, the route is considered invalid.

One way to overcome this difficulty is with the `as-override` statement, which is applied to the PE devices. The `as-override` statement replaces the CE device's AS number with that of the PE device, thus preventing the customer AS number from appearing more than once in the AS path attribute.

If a customer uses AS path prepending to make certain paths less desirable and the service provider uses AS override, each CE AS number occurrence in the AS-path is changed to the service provider AS number. For example, suppose that all customer sites use the same AS number, say 64512. If the ISP uses AS number 65534, one customer site sees the path to another site as 65534 65534. If the customer prepends 64512 on a particular path to make it less desirable, another customer site sees that path as 65534 65534 65534.

SEE ALSO

> *Example: Configuring a Layer 3 VPN with Route Reflection and AS Override*

Example: Configuring a Layer 3 VPN with Route Reflection and AS Override

IN THIS SECTION

- Requirements | 228
- Overview | 228
- Configuration | 229
- Verification | 240

Suppose that you are a service provider providing a managed MPLS-based Layer 3 VPN service. Your customer has several sites and requires BGP routing to customer edge (CE) devices at each site.

Requirements

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

Overview

IN THIS SECTION

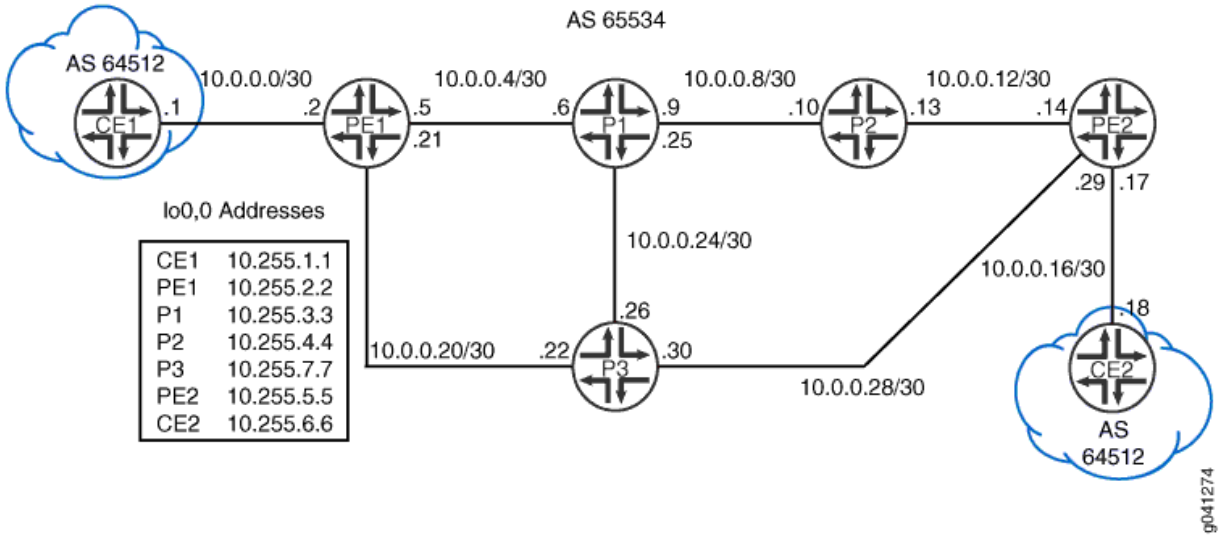
- Topology | 229

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 13 on page 229 shows the topology used in this example.

Figure 13: AS Override Topology



"CLI Quick Configuration" on page 229 shows the configuration for all of the devices in Figure 13 on page 229. The section "Step-by-Step Procedure" on page 235 describes the steps on Device PE1.

Topology

Configuration

IN THIS SECTION

- Procedure | 229

Procedure

CLI Quick Configuration

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

Device CE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.1.1/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0101.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.2
set policy-options policy-statement ToBGP term Direct from protocol direct
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.1.1
set routing-options autonomous-system 64512
```

Device P1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.6/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.25/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.3.3/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0303.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.3.3
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
```

```
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.3.3
```

Device P2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.10/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.13/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.4.4/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0404.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group Core-RRClients type internal
set protocols bgp group Core-RRClients local-address 10.255.4.4
set protocols bgp group Core-RRClients family inet-vpn unicast
set protocols bgp group Core-RRClients cluster 10.255.4.4
set protocols bgp group Core-RRClients peer-as 65534
set protocols bgp group Core-RRClients neighbor 10.255.3.3
set protocols bgp group Core-RRClients neighbor 10.255.7.7
set protocols bgp group Core-RRClients neighbor 10.255.2.2
set protocols bgp group Core-RRClients neighbor 10.255.5.5
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.4.4
set routing-options autonomous-system 65534
```

Device P3

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.22/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
```

```
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.26/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.30/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.7.7/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0707.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.7.7
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.7.7
```

Device PE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.21/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.2.2/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0202.00
set protocols mpls interface ge-1/2/2.0
set protocols mpls interface ge-1/2/1.0
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
```

```
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.2.2
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/1.0 level 2 metric 10
set protocols isis interface ge-1/2/1.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 as-override
set routing-options router-id 10.255.2.2
set routing-options autonomous-system 65534
```

Device PE2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.14/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.17/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.29/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.5.5/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0505.00
set protocols mpls interface ge-1/2/0.0
set protocols mpls interface ge-1/2/2.0
```

```
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.5.5
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/0.0 level 2 metric 10
set protocols isis interface ge-1/2/0.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/0.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/1.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 as-override
set routing-options router-id 10.255.5.5
set routing-options autonomous-system 65534
```

Device CE2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.18/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.6.6/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0606.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.17
set policy-options policy-statement ToBGP term Direct from protocol direct
```

```
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.6.6
set routing-options autonomous-system 64512
```

Step-by-Step Procedure

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

To configure AS override:

1. Configure the interfaces.

   To enable MPLS, include the protocol family on the interface so that the interface does not discard incoming MPLS traffic.

```
[edit interfaces]
user@PE1# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
user@PE1# set ge-1/2/0 unit 0 family iso
user@PE1# set ge-1/2/0 unit 0 family mpls
user@PE1# set ge-1/2/1 unit 0 family inet address 10.0.0.5/30
user@PE1# set ge-1/2/1 unit 0 family iso
user@PE1# set ge-1/2/1 unit 0 family mpls
user@PE1# set ge-1/2/2 unit 0 family inet address 10.0.0.21/30
user@PE1# set ge-1/2/2 unit 0 family iso
user@PE1# set ge-1/2/2 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.255.2.2/32
user@PE1# set lo0 unit 0 family iso address 49.0001.0010.0000.0202.00
```

2. Add the interface to the MPLS protocol to establish the control plane level connectivity.

   Set up the IGP so that the provider devices can communicate with each other.

   To establish a mechanism to distribute MPLS labels, enable LDP. Optionally, for LDP, enable forwarding equivalence class (FEC) deaggregation, which results in faster global convergence.

```
[edit protocols]
user@PE1# set mpls interface ge-1/2/2.0
user@PE1# set mpls interface ge-1/2/1.0
user@PE1# set mpls interface lo0.0
user@PE1# set mpls interface fxp0.0 disable
```

```
user@PE1# set isis interface ge-1/2/1.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/1.0 level 1 disable
user@PE1# set isis interface ge-1/2/2.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/2.0 level 1 disable
user@PE1# set isis interface fxp0.0 disable
user@PE1# set isis interface lo0.0 level 2 metric 0
user@PE1# set ldp deaggregate
user@PE1# set ldp interface ge-1/2/1.0
user@PE1# set ldp interface ge-1/2/2.0
user@PE1# set ldp interface fxp0.0 disable
user@PE1# set ldp interface lo0.0
```

3. Enable the internal BGP (IBGP) connection to peer with the RR only, using the IPv4 VPN unicast address family.

```
[edit protocols bgp group l3vpn]
user@PE1# set type internal
user@PE1# set local-address 10.255.2.2
user@PE1# set family inet-vpn unicast
user@PE1# set peer-as 65534
user@PE1# set local-as 65534
user@PE1# set neighbor 10.255.4.4
```

4. Configure the routing instance, including the `as-override` statement.

   Create the routing-Instance (VRF) on the PE device, setting up the BGP configuration to peer with Device CE1.

```
[edit routing-instances VPN-A]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 65534:1234
user@PE1# set vrf-target target:65534:1234
user@PE1# set protocols bgp group CE type external
user@PE1# set protocols bgp group CE family inet unicast
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 as-override
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@PE1# set router-id 10.255.2.2
user@PE1# set autonomous-system 65534
```

Results

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

```
user@PE1# show interfaces
ge-1/2/0 {
    unit 2 {
        family inet {
            address 10.0.0.2/30;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/1 {
    unit 5 {
        family inet {
            address 10.0.0.5/30;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 21 {
        family inet {
            address 10.0.0.21/30;
        }
        family iso;
        family mpls;
    }
}
```

```
lo0 {
    unit 0 {
        family inet {
            address 10.255.2.2/32;
        }
        family iso {
            address 49.0001.0010.0000.0202.00;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/2/2.0;
    interface ge-1/2/1.0;
    interface lo0.0;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group l3vpn {
        type internal;
        local-address 10.255.2.2;
        family inet-vpn {
            unicast;
        }
        peer-as 65534;
        local-as 65534;
        neighbor 10.255.4.4;
    }
}
isis {
    interface ge-1/2/1.0 {
        level 2 metric 10;
        level 1 disable;
    }
    interface ge-1/2/2.0 {
        level 2 metric 10;
        level 1 disable;
    }
```

```
    interface fxp0.0 {
        disable;
    }
    interface lo0.0 {
        level 2 metric 0;
    }
}
ldp {
    deaggregate;
    interface ge-1/2/1.0;
    interface ge-1/2/2.0;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
```

```
user@PE1# show routing-instances
VPN-A {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 65534:1234;
    vrf-target target:65534:1234;
    protocols {
        bgp {
            group CE {
                type external;
                family inet {
                    unicast;
                }
                neighbor 10.0.0.1 {
                    peer-as 64512;
                    as-override;
                }
            }
        }
```

```
    }
}
```

```
user@PE1# show routing-options
router-id 10.255.2.2;
autonomous-system 65534;
```

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

Verification

IN THIS SECTION

- Checking AS Path to the CE Devices | **240**
- Checking How the Route to Device CE2 Is Advertised | **241**
- Checking the Route on Device CE1 | **241**

Confirm that the configuration is working properly.

Checking AS Path to the CE Devices

Purpose

Display information on Device PE1 about the AS path attribute for the route to Device CE2's loopback interface.

Action

On Device PE1, from operational mode, enter the `show route table VPN-A.inet.0 10.255.6.6` command.

```
user@PE1> show route table VPN-A.inet.0 10.255.6.6


VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.6.6/32      *[BGP/170] 02:19:35, localpref 100, from 10.255.4.4
```

```
                    AS path: 64512 I, validation-state: unverified
                  > to 10.0.0.22 via ge-1/2/2.0, Push 300032, Push 299776(top)
```

Meaning

The output shows that Device PE1 has an AS path for 10.255.6.6/32 as coming from AS 64512.

Checking How the Route to Device CE2 Is Advertised

Purpose

Make sure the route to Device CE2 is advertised to Device CE1 as if it is coming from the MPLS core.

Action

On Device PE1, from operational mode, enter the `show route advertising-protocol bgp 10.0.0.1` command.

```
user@PE1> show route advertising-protocol bgp 10.0.0.1

VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.16/30            Self                                    I
* 10.255.1.1/32           10.0.0.1                                65534 I
* 10.255.6.6/32           Self                                    65534 I
```

Meaning

The output indicates that Device PE1 is advertising only its own AS number in the AS path.

Checking the Route on Device CE1

Purpose

Make sure that Device CE1 contains only the provider AS number in the AS path for the route to Device CE2.

Action

From operational mode, enter the `show route table inet.0 terse 10.255.6.6` command.

```
user@CE1> show route table inet.0 terse 10.255.6.6

inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.255.6.6/32      B 170        100                           65534 65534 I
  unverified                                      >10.0.0.2
```

Meaning

The output indicates that Device CE1 has a route to Device CE2. The loop issue is resolved with the use of the `as-override` statement.

One route is hidden on the CE device. This is because Junos OS does not perform a BGP split horizon. Generally, split horizon in BGP is unnecessary, because any routes that might be received back by the originator are less preferred due to AS path length (for EBGP), AS path loop detection (IBGP), or other BGP metrics. Advertising routes back to the neighbor from which they were learned has a negligible effect on the router's performance, and is the correct thing to do.

SEE ALSO

- Understanding AS Override

Example: Enabling BGP Route Advertisements

IN THIS SECTION

- Requirements | 243
- Overview | 243
- Configuration | 244

- Verification | 251

Junos OS does not advertise the routes learned from one EBGP peer back to the same external BGP (EBGP) peer. In addition, the software does not advertise those routes back to any EBGP peers that are in the same autonomous system (AS) as the originating peer, regardless of the routing instance. You can modify this behavior by including the `advertise-peer-as` statement in the configuration.

If you include the `advertise-peer-as` statement in the configuration, BGP advertises the route regardless of this check.

To restore the default behavior, include the `no-advertise-peer-as` statement in the configuration:

```
no-advertise-peer-as;
```

The route suppression default behavior is disabled if the `as-override` statement is included in the configuration. If you include both the `as-override` and `no-advertise-peer-as` statements in the configuration, the `no-advertise-peer-as` statement is ignored.

Requirements

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

NOTE: This example was updated and re-validated on Junos release 21.2R1.

Overview

IN THIS SECTION

- Topology | 244

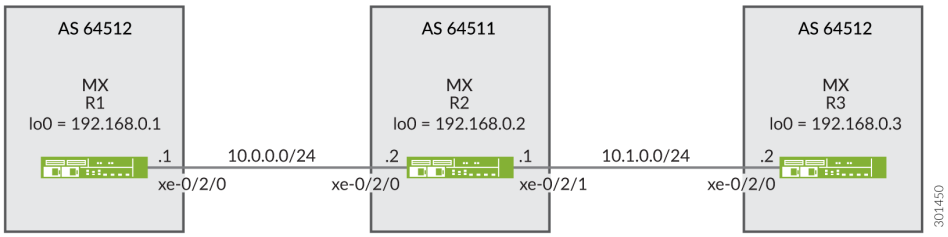
This example shows three routing devices with external BGP (EBGP) connections. Device R2 has an EBGP connection to Device R1 and another EBGP connection to Device R3. Although separated by Device R2 which is in AS 64511, Device R1 and Device R3 are in the same AS (AS 64512). Device R1 and Device R3 advertise into BGP direct routes to their own loopback interface addresses.

Device R2 receives these loopback interface routes, and the `advertise peer-as` statement allows Device R2 to advertise them. Specifically, Device R1 sends the 192.168.0.1 route to Device R2, and because Device R2 has the `advertise peer-as` configured, Device R2 can send the 192.168.0.1 route to Device R3. Likewise, Device R3 sends the 192.168.0.3 route to Device R2, and `advertise peer-as` enables Device R2 to forward the route to Device R1.

To enable Device R1 and Device R3 to accept routes that contain their own AS number in the AS path, the `loops 2` statement is required on Device R1 and Device R3.

Topology

Figure 14: BGP Topology for advertise-peer-as



Configuration

IN THIS SECTION

- CLI Quick Configuration | 244
- Procedure | 246

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 R1

```
set interfaces xe-0/2/0 description R1-to-R2
set interfaces xe-0/2/0 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 family inet unicast loops 2
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64512
```

Device R2

```
set interfaces xe-0/2/0 description R2-to-R1
set interfaces xe-0/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces xe-0/2/1 description R2-to-R3
set interfaces xe-0/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext advertise-peer-as
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 64512
set protocols bgp group ext neighbor 10.1.0.2 peer-as 64512
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64511
```

Device R3

```
set interfaces xe-0/2/0 description R3-to-R2
set interfaces xe-0/2/0 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp family inet unicast loops 2
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64512
```

Procedure

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set xe-0/2/0 description R1-to-R2
user@R1# set xe-0/2/0 unit 0 family inet address 10.0.0.1/30
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set peer-as 64511
user@R1# set neighbor 10.0.0.2
```

3. Prevent routes from Device R3 from being hidden on Device R1 by including the `loops 2` statement.

   The `loops 2` statement means that the local device's own AS number can appear in the AS path up to one time without causing the route to be hidden. The route is hidden if the local device's AS number is detected in the path two or more times.

```
[edit protocols bgp family inet unicast]
user@R1# set loops 2
```

4. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

5. Apply the export policy to the BGP peering session with Device R2.

```
[edit protocols bgp group ext]
user@R1# set export send-direct
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options ]
user@R1# set autonomous-system 64512
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set xe-0/2/0 description R2-to-R1
user@R2# set xe-0/2/0 unit 0 family inet address 10.0.0.2/30
user@R2# set xe-0/2/1 description R2-to-R3
user@R2# set xe-0/2/1 unit 0 family inet address 10.1.0.1/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure BGP.

```
[edit  protocols bgp group ext]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 64512
user@R2# set neighbor 10.1.0.2 peer-as 64512
```

3. Configure Device R2 to advertise routes learned from one EBGP peer to another EBGP peer in the same AS.

   In other words, advertise to Device R1 routes learned from Device R3 (and the reverse), even though Device R1 and Device R3 are in the same AS.

```
[edit  protocols bgp group ext]
user@R2# set advertise-peer-as
```

4. Configure a routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

5. Apply the export policy.

```
[edit protocols bgp group ext]
user@R2# set export send-direct
```

6. Configure the AS number.

```
[edit routing-options]
user@R2# set autonomous-system 64511
```

Results

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

Device R1

```
user@R1# show interfaces
xe-0/2/0 {
    description R1-to-R2;
    unit 0 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
```

```
    }
}
```

```
user@R1# show protocols
bgp {
    family inet {
        unicast {
            loops 2;
        }
    }
    group ext {
        type external;
        export send-direct;
        peer-as 64511;
        neighbor 10.0.0.2;
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show routing-options
autonomous-system 64512;
```

Device R2

```
user@R2# show interfaces
xe-0/2/0 {
    description R2-to-R1;
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
```

```
    }
}
xe-0/2/1 {
    description R2-to-R3;
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        type external;
        advertise-peer-as;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 64512;
        }
        neighbor 10.1.0.2 {
            peer-as 64512;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
```

```
    }
}
```

```
user@R2# show routing-options
autonomous-system 64511;
```

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

Verification

IN THIS SECTION

- Verifying the BGP Routes | **251**

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose

Make sure that the routing tables on Device R1 and Device R3 contain the expected routes.

Action

1. On Device R2, deactivate the `advertise-peer-as` statement in the BGP configuration.

```
[edit  protocols bgp group ext]
user@R2# deactivate advertise-peer-as
user@R2# commit
```

2. On Device R3, deactivate the `loops` statement in the BGP configuration.

```
[edit  protocols bgp family inet unicast ]
user@R3# deactivate unicast loops
user@R3# commit
```

3. On Device R1, check to see what routes are advertised to Device R2.

```
user@R1> show route advertising-protocol bgp 10.0.0.2
inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 192.168.0.1/32          Self                                    I
```

4. On Device R2, check to see what routes are received from Device R1.

```
user@R2> show route receive-protocol bgp 10.0.0.1
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  10.0.0.0/30             10.0.0.1                                64512 I
* 192.168.0.1/32          10.0.0.1                                64512 I
```

5. On Device R2, check to see what routes are advertised to Device R3.

```
user@R2> show route advertising-protocol bgp 10.1.0.2
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 10.1.0.0/30             Self                                    I
* 192.168.0.2/32          Self                                    I
```

6. On Device R2, activate the `advertise-peer-as` statement in the BGP configuration.

```
[edit  protocols bgp group ext]
user@R2# activate advertise-peer-as
user@R2# commit
```

7. On Device R2, recheck the routes that are advertised to Device R3.

```
user@R2> show route advertising-protocol bgp 10.1.0.2
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 10.1.0.0/30             Self                                    I
```

```
* 192.168.0.1/32          Self            64512 I
* 192.168.0.2/32          Self                          I
* 192.168.0.3/32          10.1.0.2        64512 I
```

8. On Device R3, check the routes that are received from Device R2.

```
user@R3> show route receive-protocol bgp 10.1.0.1
inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             10.1.0.1                                64511 I
  10.1.0.0/30             10.1.0.1                                64511 I
* 192.168.0.2/32          10.1.0.1                                64511 I
```

9. On Device R3, activate the `loops` statement in the BGP configuration.

```
[edit  protocols bgp family inet unicast ]
user@R3# activate unicast loops
user@R3# commit
```

10. On Device R3, recheck the routes that are received from Device R2.

```
user@R3> show route receive-protocol bgp 10.1.0.1
inet.0: 6 destinations, 8 routes (6 active, 0 holddown, 1 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             10.1.0.1                                64511 I
  10.1.0.0/30             10.1.0.1                                64511 I
* 192.168.0.1/32          10.1.0.1                                64511 64512 I
* 192.168.0.2/32          10.1.0.1                                64511 I
```

Meaning

First the `advertise-peer-as` statement and the `loops` statement are deactivated so that the default behavior can be examined. Device R1 sends to Device R2 a route to Device R1's loopback interface address, 192.168.0.1/32. Device R2 does not advertise this route to Device R3. After activating the `advertise-peer-as` statement, Device R2 does advertise the 192.168.0.1/32 route to Device R3. Device R3 does not accept this route until after the `loops` statement is activated.

SEE ALSO

Example: Configuring a Layer 3 VPN with Route Reflection and AS Override

Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection

BGP loop detection for a specific route uses the local autonomous system (AS) domain for the routing instance. By default, all routing instances belong to a single primary routing instance domain. Therefore, BGP loop detection uses the local ASs configured on all of the routing instances. Depending on your network configuration, this default behavior can cause routes to be looped and hidden.

To limit the local ASs in the primary routing instance, you can configure an independent AS domain for a routing instance. The independent domain is separate from the primary routing instance and keeps the AS paths of the independent domain from being shared with the AS path and the AS path attributes of other domains.

By default, independent domains use transitive path attribute 128 (attribute set) messages to tunnel the independent domain's BGP attributes through the internal BGP (IBGP) core. However, the attribute set message behavior for independent domains is undesired in many cases. If 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, you can disable the attribute set messages.

To disable attribute set messages on an independent domain, include the `independent-domain no-attrset` statement:

1. Select the routing instance that contains the independent domain you want to modify. You can select the routing instance from the following hierarchy levels:
   - `[edit routing-instances routing-instance-name]`
   - `[edit logical-systems logical-system-name routing-instances routing-instance-name]`
2. Disable attribute set messages on the independent domain.

```
[edit routing-instances instance-name]
user@host# set routing-options autonomous-system as-number independent-domain no-attrset
```

TIP: When you disable attribute set messages, we recommend that you specify the AS number of the primary routing instance. This ensures that the primary routing instance AS is treated as a local AS in the routing instance and is used for BGP loop detection.

After you specify a routing instance for an independent domain, the local ASs are only associated with that routing instance. That means BGP loop detection uses only the local ASs defined in the routing instance.

SEE ALSO

autonomous-system

independent-domain

Example: Configuring a Local AS for EBGP Sessions | **146**

Example: Ignoring the AS Path Attribute When Selecting the Best Path

IN THIS SECTION

- Requirements | **255**
- Overview | **256**
- Configuration | **257**
- Verification | **264**

If multiple BGP routes to the same destination exist, BGP selects the best path based on the route attributes of the paths. One of the route attributes that affects the best-path decision is the length of the AS paths of each route. Routes with shorter AS paths are preferred over those with longer AS paths. Although not typically practical, some scenarios might require that the AS path length be ignored in the route selection process. This example shows how to configure a routing device to ignore the AS path attribute.

Requirements

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

Overview

On externally connected routing devices, the purpose of skipping the AS path comparison might be to force an external BGP (EBGP) versus internal BGP (IBGP) decision to remove traffic from your network as soon as possible. On internally connected routing devices, you might want your IBGP-only routers to default to the local externally connected gateway. The local IBGP-only (internal) routers skip the AS path comparison and move down the decision tree to use the closest interior gateway protocol (IGP) gateway (lowest IGP metric). Doing this might be an effective way to force these routers to use a LAN connection instead of their WAN connection.

CAUTION: When you include the `as-path-ignore` statement on a routing device in your network, you might need to include it on all other BGP-enabled devices in your network to prevent routing loops and convergence issues. This is especially true for IBGP path comparisons.

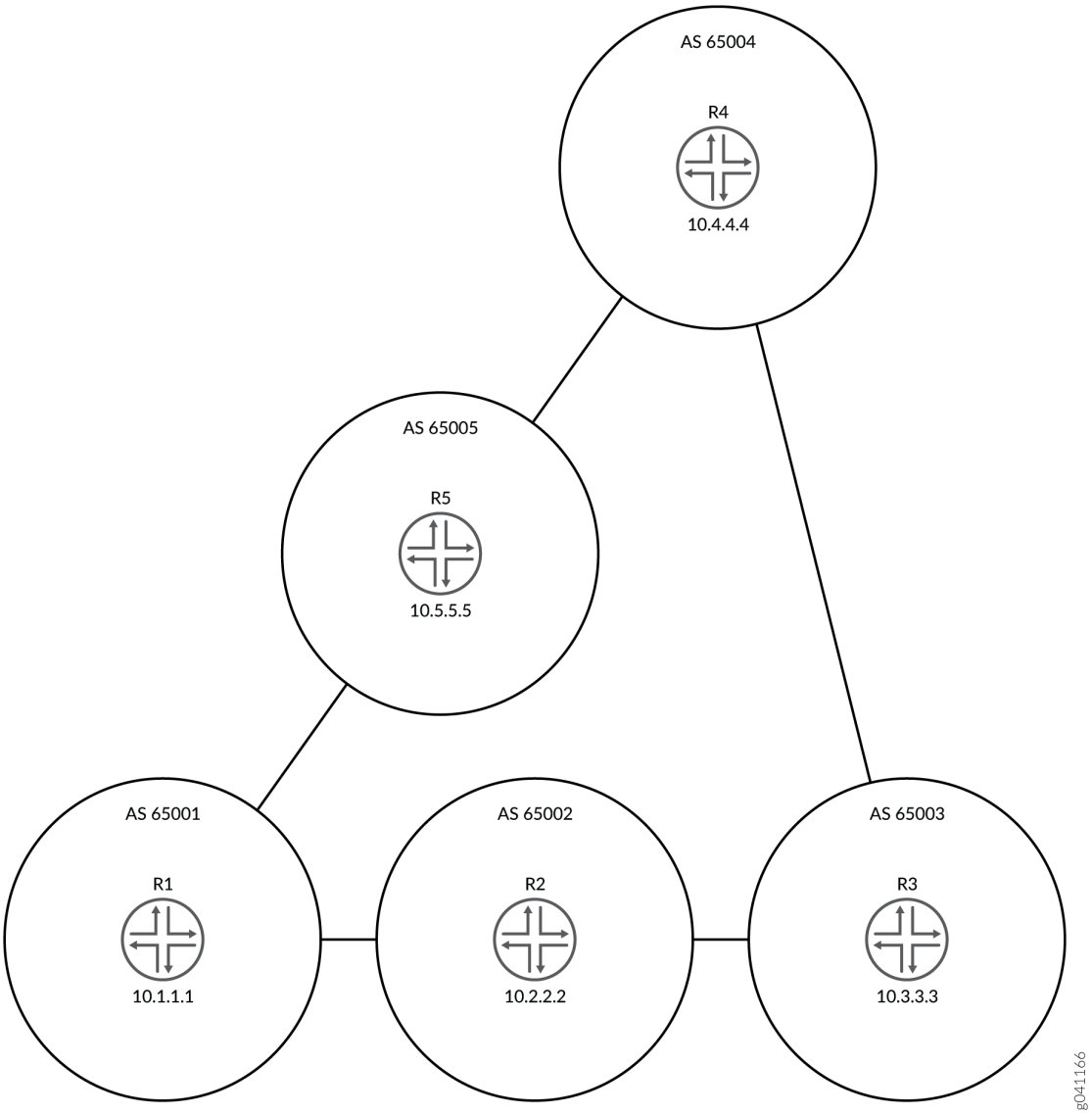
In this example, Device R2 is learning about the loopback interface address on Device R4 (10.4.4.4/32) from Device R1 and Device R3. Device R1 is advertising 10.4.4.4/32 with an AS-path of 65001 65005 65004, and Device R3 is advertising 10.4.4.4/32 with an AS-path of 65003 65004. Device R2 selects the path for 10.4.4.4/32 from Device R3 as the best path because the AS path is shorter than the AS path from Device R1.

This example modifies the BGP configuration on Device R2 so that the AS-path length is not used in the best-path selection.

Device R1 has a lower router ID (10.1.1.1) than Device R3 (10.3.3.3). If all other path selection criteria are equal (or, as in this case, ignored), the route learned from Device R1 is used. Because the AS-path attribute is being ignored, the best path is toward Device R1 because of its lower router ID value.

Figure 15 on page 257 shows the sample topology.

Figure 15: Topology for Ignoring the AS-Path length



Configuration

IN THIS SECTION

- CLI Quick Configuration | **258**
- Configuring Device R2 | **261**

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 192.168.10.1/24
set interfaces fe-1/2/1 unit 10 family inet address 192.168.50.2/24
set interfaces lo0 unit 1 family inet address 10.1.1.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext export send-local
set protocols bgp group ext neighbor 192.168.10.2 peer-as 65002
set protocols bgp group ext neighbor 192.168.50.1 peer-as 65005
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-local term 1 from protocol local
set policy-options policy-statement send-local term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.20.0/24 next-hop 192.168.10.2
set routing-options static route 192.168.30.0/24 next-hop 192.168.10.2
set routing-options static route 192.168.40.0/24 next-hop 192.168.50.1
set routing-options router-id 10.1.1.1
set routing-options autonomous-system 65001
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 192.168.10.2/24
set interfaces fe-1/2/1 unit 3 family inet address 192.168.20.2/24
set interfaces lo0 unit 2 family inet address 10.2.2.2/32
set protocols bgp path-selection as-path-ignore
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext export send-local
set protocols bgp group ext neighbor 192.168.10.1 peer-as 65001
set protocols bgp group ext neighbor 192.168.20.1 peer-as 65003
set policy-options policy-statement send-direct term 1 from protocol direct
```

```
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-local term 1 from protocol local
set policy-options policy-statement send-local term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.50.0/24 next-hop 192.168.10.1
set routing-options static route 192.168.40.0/24 next-hop 192.168.10.1
set routing-options static route 192.168.30.0/24 next-hop 192.168.20.1
set routing-options router-id 10.2.2.2
set routing-options autonomous-system 65002
```

Device R3

```
set interfaces fe-1/2/0 unit 4 family inet address 192.168.20.1/24
set interfaces fe-1/2/1 unit 5 family inet address 192.168.30.1/24
set interfaces lo0 unit 3 family inet address 10.3.3.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext export send-local
set protocols bgp group ext neighbor 192.168.20.2 peer-as 65002
set protocols bgp group ext neighbor 192.168.30.2 peer-as 65004
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-local term 1 from protocol local
set policy-options policy-statement send-local term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.10.0/24 next-hop 192.168.20.2
set routing-options static route 192.168.50.0/24 next-hop 192.168.20.2
set routing-options static route 192.168.40.0/24 next-hop 192.168.30.2
set routing-options router-id 10.3.3.3
set routing-options autonomous-system 65003
```

Device R4

```
set interfaces fe-1/2/0 unit 6 family inet address 192.168.30.2/24
set interfaces fe-1/2/1 unit 7 family inet address 192.168.40.1/24
set interfaces lo0 unit 4 family inet address 10.4.4.4/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
```

```
set protocols bgp group ext export send-static
set protocols bgp group ext export send-local
set protocols bgp group ext neighbor 192.168.30.1 peer-as 65003
set protocols bgp group ext neighbor 192.168.40.2 peer-as 65005
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-local term 1 from protocol local
set policy-options policy-statement send-local term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.10.0/24 next-hop 192.168.40.2
set routing-options static route 192.168.50.0/24 next-hop 192.168.40.2
set routing-options static route 192.168.40.0/24 next-hop 192.168.30.1
set routing-options router-id 10.4.4.4
set routing-options autonomous-system 65004
```

Device R5

```
set interfaces fe-1/2/0 unit 8 family inet address 192.168.40.2/24
set interfaces fe-1/2/1 unit 9 family inet address 192.168.50.1/24
set interfaces lo0 unit 5 family inet address 10.5.5.5/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext export send-local
set protocols bgp group ext neighbor 192.168.40.1 peer-as 65004
set protocols bgp group ext neighbor 192.168.50.2 peer-as 65001
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-local term 1 from protocol local
set policy-options policy-statement send-local term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.10.0/24 next-hop 192.168.50.2
set routing-options static route 192.168.20.0/24 next-hop 192.168.50.2
set routing-options static route 192.168.30.0/24 next-hop 192.168.40.1
set routing-options router-id 10.5.5.5
set routing-options autonomous-system 65005
```

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 2 family inet address 192.168.10.2/24
user@R2# set fe-1/2/1 unit 3 family inet address 192.168.20.2/24
user@R2# set lo0 unit 2 family inet address 10.2.2.2/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set export send-direct
user@R2# set export send-static
user@R2# set export send-local
user@R2# set neighbor 192.168.10.1 peer-as 65001
user@R2# set neighbor 192.168.20.1 peer-as 65003
```

3. Configure the autonomous system (AS) path attribute to be ignored in the Junos OS path selection algorithm.

```
[edit protocols bgp]
user@R2# set path-selection as-path-ignore
```

4. Configure the routing policy.

```
[edit policy-options]
user@R2# set policy-statement send-direct term 1 from protocol direct
user@R2# set policy-statement send-direct term 1 then accept
user@R2# set policy-statement send-local term 1 from protocol local
user@R2# set policy-statement send-local term 1 then accept
```

```
user@R2# set policy-statement send-static term 1 from protocol static
user@R2# set policy-statement send-static term 1 then accept
```

5. Configure some static routes.

```
[edit routing-options static]
user@R2# set route 192.168.50.0/24 next-hop 192.168.10.1
user@R2# set route 192.168.40.0/24 next-hop 192.168.10.1
user@R2# set route 192.168.30.0/24 next-hop 192.168.20.1
```

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

```
[edit routing-options]
user@R2# set router-id 10.2.2.2
user@R2# set autonomous-system 65002
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            address 192.168.10.2/24;
        }
    }
}
fe-1/2/1 {
    unit 3 {
        family inet {
            address 192.168.20.2/24;
        }
    }
}
lo0 {
    unit 2 {
```

```
        family inet {
            address 10.2.2.2/32;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement send-local {
    term 1 {
        from protocol local;
        then accept;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@R2# show protocols
bgp {
    path-selection as-path-ignore;
    group ext {
        type external;
        export [ send-direct send-static send-local ];
        neighbor 192.168.10.1 {
            peer-as 65001;
        }
        neighbor 192.168.20.1 {
            peer-as 65003;
        }
```

```
    }
}
```

```
user@R2# show routing-options
static {
    route 192.168.50.0/24 next-hop 192.168.10.1;
    route 192.168.40.0/24 next-hop 192.168.10.1;
    route 192.168.30.0/24 next-hop 192.168.20.1;
}
router-id 10.2.2.2;
autonomous-system 65002;
```

If you are done configuring the device, enter **commit** from configuration mode. Repeat the configuration on the other devices in the network, changing the interface names and IP addresses, as needed.

Verification

IN THIS SECTION

- Checking the Neighbor Status | **264**

Confirm that the configuration is working properly.

Checking the Neighbor Status

Purpose

Make sure that from Device R2, the active path to get to AS 4 is through AS 65001 and AS 65005, not through AS 65003.

NOTE: To verify the functionality of the `as-path-ignore` statement, you might need to run the `restart routing` command to force reevaluation of the active path. This is because for BGP, if both paths are external, the Junos OS behavior is to prefer the currently active path. This behavior helps to minimize route-flapping. Use caution when restarting the routing protocol process in a production network.

Action

From operational mode, enter the `restart routing` command.

```
user@R2> restart routing
Routing protocols process started, pid 49396
```

From operational mode, enter the `show route 10.4.4.4 protocol bgp` command.

```
user@R2> show route 10.4.4.4 protocol bgp
inet.0: 12 destinations, 25 routes (12 active, 0 holddown, 4 hidden)
+ = Active Route, - = Last Active, * = Both

10.4.4.4/32         *[BGP/170] 00:00:12, localpref 100
                      AS path: 65001 65005 65004 I
                    > to 192.168.10.1 via fe-1/2/0.2
                    [BGP/170] 00:00:08, localpref 100
                      AS path: 65003 65004 I
                    > to 192.168.20.1 via fe-1/2/1.3
```

Meaning

The asterisk (*) is next to the path learned from R1, meaning that this is the active path. The AS path for the active path is 65001 65005 65004, which is longer than the AS path (65003 65004) for the nonactive path learned from Router R3.

SEE ALSO

Understanding BGP Path Selection | 12

Understanding Private AS Number Removal from AS Paths

By default, when BGP advertises AS paths to remote systems, it includes all AS numbers, including private AS numbers. You can configure the software so that it removes private AS numbers from AS paths. Doing this is useful when any of the following circumstances are true:

- A remote AS for which you provide connectivity is multihomed, but only to the local AS.

- The remote AS does not have an officially allocated AS number.
- It is not appropriate to make the remote AS a confederation member AS of the local AS.

Most companies acquire their own AS number. Some companies also use private AS numbers to connect to their public AS network. These companies might use a different private AS number for each region in which their company does business. In any implementation, announcing a private AS number to the Internet must be avoided. Service providers can use the `remove-private` statement to prevent advertising private AS numbers to the Internet.

In an enterprise scenario, suppose that you have multiple AS numbers in your company, some of which are private AS numbers, and one with a public AS number. The one with a public AS number has a direct connection to the service provider. In the AS that connects directly to the service provider, you can use the `remove-private` statement to filter out any private AS numbers in the advertisements that are sent to the service provider.

The AS numbers are stripped from the AS path starting at the left end of the AS path (the end where AS paths have been most recently added). The routing device stops searching for private ASs when it finds the first nonprivate AS or a peer's private AS. If the AS path contains the AS number of the external BGP (EBGP) neighbor, BGP does not remove the private AS number.

NOTE: As of Junos OS 10.0R2 and later, if there is a need to send prefixes to an EBGP peer that has an AS number that matches an AS number in the AS path, consider using the `as-override` statement instead of the `remove-private` statement.

The operation takes place after any confederation member ASs have already been removed from the AS path, if applicable.

The software is preconfigured with knowledge of the set of AS numbers that is considered private, a range that is defined in the Internet Assigned Numbers Authority (IANA) assigned numbers document. The set of 16 bit AS numbers reserved as private are in the range from 64,512 through 65,534, inclusive. The 32 bit AS numbers reserved as private are in the range from 4,200,000,000 through 4,294,967,294 inclusive.

SEE ALSO

Example: Removing Private AS Numbers from AS Paths | 267

Example: Removing Private AS Numbers from AS Paths

IN THIS SECTION

- Requirements | 267
- Overview | 267
- Configuration | 268
- Verification | 272

This example demonstrates the removal of a private AS number from the advertised AS path to avoid announcing the private AS number to the Internet.

Requirements

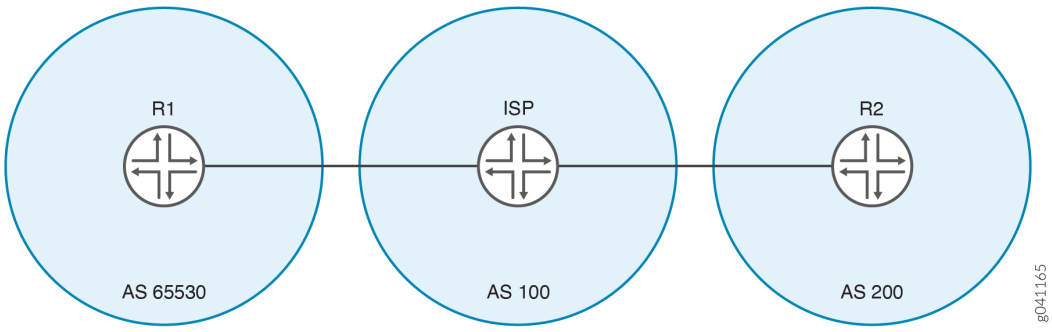
No special configuration beyond device initialization is required before you configure this example.

Overview

Service providers and enterprise networks use the `remove-private` statement to prevent advertising private AS numbers to the Internet. The `remove-private` statement works in the outbound direction. You configure the `remove-private` statement on a device that has a public AS number and that is connected to one or more devices that have private AS numbers. Generally, you would not configure this statement on a device that has a private AS number.

Figure 16 on page 267 shows the sample topology.

Figure 16: Topology for Removing a Private AS from the Advertised AS Path



In this example, Device R1 is connected to its service provider using private AS number 65530. The example shows the `remove-private` statement configured on Device ISP to prevent Device R1's private AS number from being announced to Device R2. Device R2 sees only the AS number of the service provider.

NOTE: Adding or deleting the BGP option `remove-private` will cause the affected BGP peering sessions to flap.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **268**
- Device ISP | **269**

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 192.168.10.1/24
set interfaces lo0 unit 1 family inet address 10.10.10.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 100
set protocols bgp group ext neighbor 192.168.10.10
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.20.0/24 next-hop 192.168.10.10
set routing-options autonomous-system 65530
```

Device ISP

```
set interfaces fe-1/2/0 unit 2 family inet address 192.168.10.10/24
set interfaces fe-1/2/1 unit 3 family inet address 192.168.20.20/24
set interfaces lo0 unit 2 family inet address 10.10.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext neighbor 192.168.10.1 peer-as 65530
set protocols bgp group ext neighbor 192.168.20.1 remove-private
set protocols bgp group ext neighbor 192.168.20.1 peer-as 200
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 4 family inet address 192.168.20.1/24
set interfaces lo0 unit 3 family inet address 10.10.20.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 100
set protocols bgp group ext neighbor 192.168.20.20
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 192.168.10.0/24 next-hop 192.168.20.20
set routing-options autonomous-system 200
```

Device ISP

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 Junos OS CLI User Guide.

To configure Device ISP:

1. Configure the interfaces.

```
[edit interfaces]
user@ISP# set fe-1/2/0 unit 2 family inet address 192.168.10.10/24
user@ISP# set fe-1/2/1 unit 3 family inet address 192.168.20.20/24
user@ISP# set lo0 unit 2 family inet address 10.10.0.1/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@ISP# set type external
user@ISP# set neighbor 192.168.10.1 peer-as 65530
user@ISP# set neighbor 192.168.20.1 peer-as 200
```

3. For the neighbor in autonomous system (AS) 200 (Device R2), remove private AS numbers from the advertised AS paths.

```
[edit protocols bgp group ext]
user@ISP# set neighbor 192.168.20.1 remove-private
```

4. Configure the AS number.

```
[edit routing-options]
user@ISP# set autonomous-system 100
```

Results

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

```
user@ISP# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            address 192.168.10.10/24;
        }
```

```
    }
}
fe-1/2/1 {
    unit 3 {
        family inet {
            address 192.168.20.20/24;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 10.10.0.1/32;
        }
    }
}
```

```
user@ISP# show protocols
bgp {
    group ext {
        type external;
        neighbor 192.168.10.1 {
            peer-as 65530;
        }
        neighbor 192.168.20.1 {
            remove-private;
            peer-as 200;
        }
    }
}
```

```
user@ISP# show routing-options
autonomous-system 100;
```

If you are done configuring the device, enter **commit** from configuration mode. Repeat the configuration on Device R1 and Device R2, changing the interface names and IP address, as needed, and adding the routing policy configuration.

Verification

IN THIS SECTION

- Checking the Neighbor Status | 272
- Checking the Routing Tables | 273
- Checking the AS Path When the remove-private Statement Is Deactivated | 275

Confirm that the configuration is working properly.

Checking the Neighbor Status

Purpose

Make sure that Device ISP has the **remove-private** setting enabled in its neighbor session with Device R2.

Action

From operational mode, enter the `show bgp neighbor 192.168.20.1` command.

```
user@ISP> show bgp neighbor 192.168.20.1
Peer: 192.168.20.1+179 AS 200  Local: 192.168.20.20+60216 AS 100
  Type: External    State: Established    Flags: <ImportEval Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference RemovePrivateAS PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.10.20.1      Local ID: 10.10.0.1         Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/2/1.3
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 200)
  Peer does not support Addpath
  Table inet.0 Bit: 10001
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            3
    Accepted prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          1
  Last traffic (seconds): Received 10   Sent 16   Checked 55
  Input messages:  Total 54     Updates 3       Refreshes 0     Octets 1091
  Output messages: Total 54     Updates 1       Refreshes 0     Octets 1118
  Output Queue[0]: 0
```

Meaning

The `RemovePrivateAS` option shows that Device ISP has the expected setting.

Checking the Routing Tables

Purpose

Make sure that the devices have the expected routes and AS paths.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.20.1/32      *[BGP/170] 00:28:57, localpref 100
```

```
                    AS path: 100 200 I
                  > to 192.168.10.10 via fe-1/2/0.1
```

```
user@ISP> show route protocol bgp

inet.0: 7 destinations, 11 routes (7 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.1/32      *[BGP/170] 00:29:40, localpref 100
                      AS path: 65530 I
                    > to 192.168.10.1 via fe-1/2/0.2
10.10.20.1/32      *[BGP/170] 00:29:36, localpref 100
                      AS path: 200 I
                    > to 192.168.20.1 via fe-1/2/1.3
192.168.10.0/24     [BGP/170] 00:29:40, localpref 100
                      AS path: 65530 I
                    > to 192.168.10.1 via fe-1/2/0.2
192.168.20.0/24     [BGP/170] 00:29:36, localpref 100
                      AS path: 200 I
                    > to 192.168.20.1 via fe-1/2/1.3
```

```
user@R2> show route protocol bgp
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.1/32      *[BGP/170] 00:29:53, localpref 100
                      AS path: 100 I
                    > to 192.168.20.20 via fe-1/2/0.4
```

Meaning

Device ISP has the private AS number 65530 in its AS path to Device R1. However, Device ISP does not advertise this private AS number to Device R2. This is shown in the routing table of Device R2. Device R2's path to Device R1 contains only the AS number for Device ISP.

Checking the AS Path When the remove-private Statement Is Deactivated

Purpose

Verify that without the `remove-private` statement, the private AS number appears in Device R2's routing table.

Action

From configuration mode on Device ISP, enter the `deactivate remove-private` command and then recheck the routing table on Device R2.

```
[protocols bgp group ext neighbor 192.168.20.1]
user@ISP# deactivate remove-private
user@ISP# commit
```

```
user@R2> show route protocol bgp
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.1/32      *[BGP/170] 00:00:54, localpref 100
                      AS path: 100 65530 I
                    > to 192.168.20.20 via fe-1/2/0.4
```

Meaning

Private AS number 65530 appears in Device R2's AS path to Device R1.

SEE ALSO

Understanding Private AS Number Removal from AS Paths | 265

Change History Table

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

Release	Description
20.2R1	Starting in Release 20.2R1, Junos OS supports the translation of AIGP metric to MED. You can enable this feature when you want the MED to carry the end to end AIGP metric value, which is used to choose the best path.

Local Preference for BGP Routes

IN THIS SECTION

- Understanding Route Preference Values (Administrative Distance) | **276**
- Example: Configuring the Preference Value for BGP Routes | **279**
- Example: Using Routing Policy to Set a Preference Value for BGP Routes | **288**
- Understanding the Local Preference Metric for Internal BGP Routes | **296**
- Example: Configuring the Local Preference Value for BGP Routes | **297**
- Example: Configuring BGP to Advertise Inactive Routes | **316**

Understanding Route Preference Values (Administrative Distance)

The Junos OS routing protocol process assigns a default preference value (also known as an *administrative distance*) to each route that the routing table receives. The default value depends on the source of the route. The preference value is a value from 0 through 4,294,967,295 (2^{32} – 1), with a lower value indicating a more preferred route. Table 4 on page 276 lists the default preference values.

Table 4: Default Route Preference Values

How Route Is Learned	Default Preference	Statement to Modify Default Preference
Directly connected network	0	–

Table 4: Default Route Preference Values *(Continued)*

How Route Is Learned	Default Preference	Statement to Modify Default Preference
System routes	4	–
Static and Static LSPs	5	*static*
ARI-TS	5	ARI-TS preference. Starting in Junos OS Release 22.2R1, ARI routes are installed as ARI-TS protocol routes instead of static routes as installed in the earlier Junos OS releases.
Static LSPs	6	MPLS preference **NOTE**: In Junos OS Releases prior to 10.4, if you configure a static MPLS LSP using the `static-path` statement, the default preference value is 5. Starting in Junos OS Release 10.4, if you configure a `static-label-switched-path` the default preference value is 6. The previous configuration statement `static-path` is hidden in Junos OS Release 10.4 and later releases.
RSVP-signaled LSPs	7	RSVP `preference` as described in the MPLS Applications User Guide
SR-TE	8	*SR-TE preference*
LDP-signaled LSPs	9	LDP `preference`, as described in the MPLS Applications User Guide
OSPF internal route	10	OSPF preference
OSPF SR route	10	Labelled OSPF preference
access-internal route	12	–

Table 4: Default Route Preference Values *(Continued)*

How Route Is Learned	Default Preference	Statement to Modify Default Preference
access route	13	–
IS-IS SR route	14	Labelled IS-IS preference
IS-IS Level 1 internal route	15	IS-IS preference
IS-IS Level 2 internal route	18	IS-IS preference
Redirects	30	–
Kernel	40	–
SNMP	50	–
Router discovery	55	–
RIP	100	RIP preference
RIPng	100	RIPng preference
PIM	105	Junos OS Multicast Protocols User Guide
DVMRP	110	Junos OS Multicast Protocols User Guide
Aggregate	130	*aggregate*
OSPF AS external routes	150	OSPF external-preference
IS-IS Level 1 external route	160	IS-IS external-preference

Table 4: Default Route Preference Values *(Continued)*

How Route Is Learned	Default Preference	Statement to Modify Default Preference
IS-IS Level 2 external route	165	IS-IS external-preference
BGP	170	BGP preference, export, import
MSDP	175	Junos OS Multicast Protocols User Guide
EVPN Type-5	170	EVPN User Guide

In general, the narrower the scope of the statement, the higher precedence its preference value is given, but the smaller the set of routes it affects. To modify the default preference value for routes learned by routing protocols, you generally apply routing policy when configuring the individual routing protocols. You also can modify some preferences with other configuration statements, which are indicated in the table.

SEE ALSO

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configuring the Preference Value for BGP Routes

IN THIS SECTION

- Requirements | **280**
- Overview | **280**
- Configuration | **282**
- Verification | **286**

This example shows how to specify the preference for routes learned from BGP. Routing information can be learned from multiple sources. To break ties among equally specific routes learned from multiple sources, each source has a preference value. Routes that are learned through explicit administrative action, such as static routes, are preferred over routes learned from a routing protocol, such as BGP or OSPF. This concept is called *administrative distance* by some vendors.

Requirements

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

Overview

IN THIS SECTION

- Topology | **281**

Routing information can be learned from multiple sources, such as through static configuration, BGP, or an interior gateway protocol (IGP). When Junos OS determines a route's preference to become the active route, it selects the route with the lowest preference as the active route and installs this route into the forwarding table. By default, the routing software assigns a preference of 170 to routes that originated from BGP. Of all the routing protocols, BGP has the highest default preference value, which means that routes learned by BGP are the least likely to become the active route.

Some vendors have a preference (distance) of 20 for external BGP (EBGP) and a distance of 200 for internal BGP (IGBP). Junos OS uses the same value (170) for both EBGP and IBGP. However, this difference between vendors has no operational impact because Junos OS always prefers EBGP routes over IBGP routes.

Another area in which vendors differ is in regard to IGP distance compared to BGP distance. For example, some vendors assign a distance of 110 to OSPF routes. This is higher than the EBGP distance of 20 , and results in the selection of an EBGP route over an equivalent OSPF route. In the same scenario, Junos OS chooses the OSPF route, because of the default preference 10 for an internal OSPF route and 150 for an external OSPF route, which are both lower than the 170 preference assigned to all BGP routes.

In a multivendor environment, you might want to change the preference value for BGP routes so that Junos OS chooses an EBGP route instead of an OSPF route. To accomplish this goal, one option is to include the `preference` statement in the EBGP configuration. To modify the default BGP preference value, include the `preference` statement, specifying a value from 0 through 4,294,967,295 (2^{32} – 1).

TIP: Another way to achieve multivendor compatibility is to include the `advertise-inactive` statement in the EBGP configuration. This causes the routing table to export to BGP the best route learned by BGP even if Junos OS did not select it to be an active route. By default, BGP stores the route information it receives from update messages in the Junos OS routing table, and the routing table exports only active routes into BGP, which BGP then advertises to its peers. The `advertise-inactive` statement causes Junos OS to advertise the best BGP route that is inactive because of IGP preference. When you use the `advertise-inactive` statement, the Junos OS device uses the OSPF route for forwarding, and the other vendor's device uses the EBGP route for forwarding. However, from the perspective of an EBGP peer in a neighboring AS, both vendors' devices appear to behave the same way.

Topology

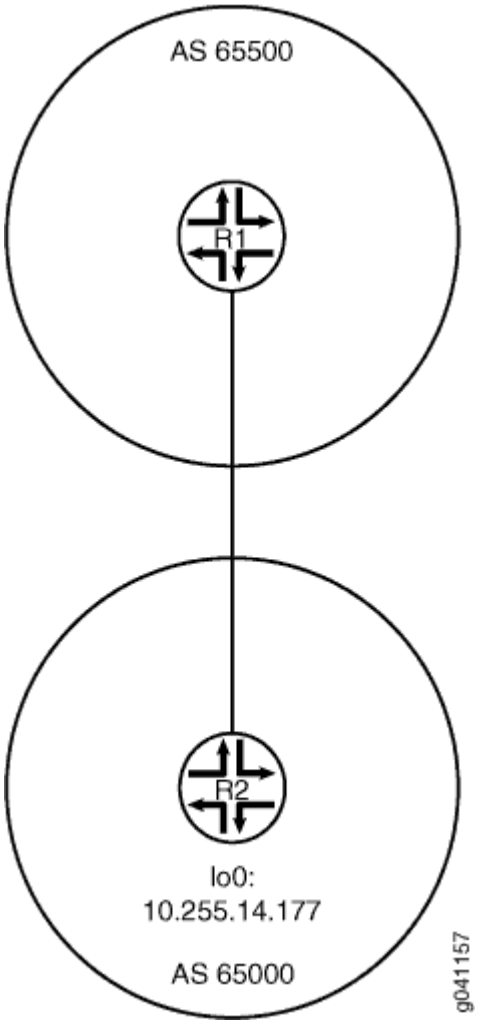
In the sample network, Device R1 and Device R2 have EBGP routes to each other and also OSPF routes to each other.

This example shows the routing tables in the following cases:

- Accept the default preference values of 170 for BGP and 10 for OSPF.
- Change the BGP preference to 8.

Figure 17 on page 282 shows the sample network.

Figure 17: BGP Preference Value Topology



Configuration

IN THIS SECTION

- CLI Quick Configuration | **283**
- Procedure | **284**

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 R1

```
set interfaces fe-1/2/0 unit 4 family inet address 1.12.0.1/30
set interfaces lo0 unit 2 family inet address 10.255.71.24/32
set protocols bgp export send-direct
set protocols bgp group ext type external
set protocols bgp group ext preference 8
set protocols bgp group ext peer-as 65000
set protocols bgp group ext neighbor 1.12.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.4
set protocols ospf area 0.0.0.0 interface 10.255.71.24
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 65500
```

Device R2

```
set interfaces fe-1/2/0 unit 6 family inet address 1.12.0.2/30
set interfaces lo0 unit 3 family inet address 10.255.14.177/32
set protocols bgp export send-direct
set protocols bgp group ext type external
set protocols bgp group ext peer-as 65500
set protocols bgp group ext neighbor 1.12.0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/0.6
set protocols ospf area 0.0.0.0 interface 10.255.14.177
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 65000
```

Procedure

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 4 family inet address 1.12.0.1/30
user@R1# set lo0 unit 2 family inet address 10.255.71.24/32
```

2. Configure the local autonomous system.

```
[edit routing-options]
user@R1# set autonomous-system 65500
```

3. Configure the external peering with Device R2.

```
[edit protocols bgp]
user@R1# set export send-direct
user@R1# set group ext type external
user@R1# set group ext preference 8
user@R1# set group ext peer-as 65000
user@R1# set group ext neighbor 1.12.0.2
```

4. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.4
user@R1# set interface 10.255.71.24
```

5. Configure the routing policy.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 4 {
        family inet {
            address 1.12.0.1/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 10.255.71.24/32;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show protocols
protocols {
```

```
    bgp {
        export send-direct;
        group ext {
            type external;
            preference 8;
            peer-as 65000;
            neighbor 1.12.0.2;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface fe-1/2/0.4;
            interface 10.255.71.24;
        }
    }
}
```

```
user@R1# show routing-options
autonomous-system 65500;
```

If you are done configuring the device, enter **commit** from configuration mode.
Repeat these steps on Device R2.

Verification

IN THIS SECTION

- Verifying the Preference | 286

Confirm that the configuration is working properly.

Verifying the Preference

Purpose

Make sure that the routing tables on Device R1 and Device R2 reflect the fact that Device R1 is using the configured EBGP preference of 8, and Device R2 is using the default EBGP preference of 170.

Action

From operational mode, enter the `show route` command.

```
user@R1> show route
inet.0: 5 destinations, 7 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.12.0.0/30        *[Direct/0] 3d 07:03:01
                    > via fe-1/2/0.4
                    [BGP/8] 01:04:49, localpref 100
                      AS path: 65000 I
                    > to 1.12.0.2 via fe-1/2/0.4
1.12.0.1/32        *[Local/0] 3d 07:03:01
                      Local via fe-1/2/0.4
10.255.14.177/32   *[BGP/8] 01:04:49, localpref 100
                      AS path: 65000 I
                    > to 1.12.0.2 via fe-1/2/0.4
                    [OSPF/10] 3d 07:02:16, metric 1
                    > to 1.12.0.2 via fe-1/2/0.4
10.255.71.24/32    *[Direct/0] 3d 07:03:01
                    > via lo0.2
224.0.0.5/32       *[OSPF/10] 5d 03:42:16, metric 1
                      MultiRecv
```

```
user@R2> show route
inet.0: 5 destinations, 7 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.12.0.0/30        *[Direct/0] 3d 07:03:30
                    > via fe-1/2/0.6
                    [BGP/170] 00:45:36, localpref 100
                      AS path: 65500 I
                    > to 1.12.0.1 via fe-1/2/0.6
1.12.0.2/32        *[Local/0] 3d 07:03:30
                      Local via fe-1/2/0.6
10.255.14.177/32   *[Direct/0] 3d 07:03:30
                    > via lo0.3
10.255.71.24/32    *[OSPF/10] 3d 07:02:45, metric 1
                    > to 1.12.0.1 via fe-1/2/0.6
                    [BGP/170] 00:45:36, localpref 100
```

```
                    AS path: 65500 I
                  > to 1.12.0.1 via fe-1/2/0.6
224.0.0.5/32      *[OSPF/10] 5d 03:42:45, metric 1
                    MultiRecv
```

Meaning

The output shows that on Device R1, the active path to Device R2's loopback interface (10.255.14.177/32) is a BGP route. The output also shows that on Device R2, the active path to Device R1's loopback interface (10.255.71.24/32) is an OSPF route.

SEE ALSO

Route Preferences Overview

Understanding External BGP Peering Sessions | **26**

BGP Configuration Overview | **24**

Example: Using Routing Policy to Set a Preference Value for BGP Routes

IN THIS SECTION

- Requirements | **289**
- Overview | **289**
- Configuration | **290**
- Verification | **295**

This example shows how to use routing policy to set the preference for routes learned from BGP. Routing information can be learned from multiple sources. To break ties among equally specific routes learned from multiple sources, each source has a preference value. Routes that are learned through explicit administrative action, such as static routes, are preferred over routes learned from a routing protocol, such as BGP or OSPF. This concept is called *administrative distance* by some vendors.

Requirements

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

Overview

IN THIS SECTION

- Topology | 289

Routing information can be learned from multiple sources, such as through static configuration, BGP, or an interior gateway protocol (IGP). When Junos OS determines a route's preference to become the active route, it selects the route with the lowest preference as the active route and installs this route into the forwarding table. By default, the routing software assigns a preference of 170 to routes that originated from BGP. Of all the routing protocols, BGP has the highest default preference value, which means that routes learned by BGP are the least likely to become the active route.

Some vendors have a preference (distance) of 20 for external BGP (EBGP) and a distance of 200 for internal BGP (IGBP). Junos OS uses the same value (170) for both EBGP and IBGP. However, this difference between vendors has no operational impact because Junos OS always prefers EBGP routes over IBGP routes.

Another area in which vendors differ is in regard to IGP distance compared to BGP distance. For example, some vendors assign a distance of 110 to OSPF routes. This is higher than the EBGP distance of 20, and results in the selection of an EBGP route over an equivalent OSPF route. In the same scenario, Junos OS chooses the OSPF route, because of the default preference 10 for an internal OSPF route and 150 for an external OSPF route, which are both lower than the 170 preference assigned to all BGP routes.

This example shows a routing policy that matches routes from specific next hops and sets a preference. If a route does not match the first term, it is evaluated by the second term.

Topology

In the sample network, Device R1 and Device R3 have EBGP sessions with Device R2.

On Device R2, an import policy takes the following actions:

- For routes received through BGP from next-hop 10.0.0.1 (Device R1), set the route preference to 10.
- For routes received through BGP from next-hop 10.1.0.2 (Device R3), set the route preference to 15.

Figure 18 on page 290 shows the sample network.

Figure 18: BGP Preference Value Topology

"CLI Quick Configuration" on page 290 shows the configuration for all of the devices in Figure 18 on page 290.

The section "No Link Title" on page 292 describes the steps on Device R2.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 290
- Procedure | 292

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 R1

```
set interfaces fe-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 ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
```

```
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext import set-preference
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement set-preference term term1 from protocol bgp
set policy-options policy-statement set-preference term term1 from next-hop 10.0.0.1
set policy-options policy-statement set-preference term term1 then preference 10
set policy-options policy-statement set-preference term term2 from protocol bgp
set policy-options policy-statement set-preference term term2 from next-hop 10.1.0.2
set policy-options policy-statement set-preference term term2 then preference 15
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 300
```

Procedure

Step-by-Step Procedure

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

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the local autonomous system.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

3. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

4. Configure the routing policy that changes the preference of received routes.

```
[edit policy-options policy-statement set-preference]
user@R2# set term term1 from protocol bgp
user@R2# set term term1 from next-hop 10.0.0.1
user@R2# set term term1 then preference 10
user@R2# set term term2 from protocol bgp
user@R2# set term term2 from next-hop 10.1.0.2
user@R2# set term term2 then preference 15
```

5. Configure the external peering with Device R2.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set export send-direct
user@R2# set neighbor 10.0.0.1 peer-as 100
user@R2# set neighbor 10.1.0.2 peer-as 300
```

6. Apply the `set-preference` policy as an import policy.

   This affects Device R2's routing table and has no impact on Device R1 and Device R3.

```
[edit protocols bgp group ext]
user@R2# set import set-preference
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
lo0 {
    unit 0{
        family inet {
            address 192.168.0.2/32;
```

```
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        type external;
        import set-preference;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
        neighbor 10.1.0.2 {
            peer-as 300;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement set-preference {
    term term1 {
        from {
            protocol bgp;
            next-hop 10.0.0.1;
        }
        then {
            preference 10;
        }
    }
    term term2 {
        from {
            protocol bgp;
            next-hop 10.1.0.2;
```

```
        }
        then {
            preference 15;
        }
    }
}
```

```
user@R2# show routing-options
autonomous-system 200;
```

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

Verification

IN THIS SECTION

- Verifying the Preference | **295**

Confirm that the configuration is working properly.

Verifying the Preference

Purpose

Make sure that the routing tables on Device R1 and Device R2 reflect the fact that Device R1 is using the configured EBGP preference of 8, and Device R2 is using the default EBGP preference of 170.

Action

From operational mode, enter the `show route protocols bgp` command.

```
user@R2> show route protocols bgp
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/30         [BGP/10] 04:42:23, localpref 100
                      AS path: 100 I, validation-state: unverified
```

```
                    > to 10.0.0.1 via fe-1/2/0.0
10.1.0.0/30         [BGP/15] 04:42:23, localpref 100
                      AS path: 300 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
192.168.0.1/32     *[BGP/10] 04:42:23, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
192.168.0.3/32     *[BGP/15] 04:42:23, localpref 100
                      AS path: 300 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
```

Meaning

The output shows that on Device R2, the preference values have been changed to 15 for routes learned from Device R3, and the preference values have been changed to 10 for routes learned from Device R1.

SEE ALSO

- Route Preferences Overview
- Understanding External BGP Peering Sessions

Understanding the Local Preference Metric for Internal BGP Routes

Internal BGP (IBGP) sessions use a metric called the *local preference*, which is carried in IBGP update packets in the path attribute LOCAL_PREF. When an autonomous system (AS) has multiple routes to another AS, the local preference indicates the degree of preference for one BGP route over the other BGP routes. The BGP route with the highest local preference value is preferred.

The LOCAL_PREF path attribute is always advertised to IBGP peers and to neighboring confederations. It is never advertised to external BGP (EBGP) peers. The default behavior is to not modify the LOCAL_PREF path attribute if it is present.

The default LOCAL_PREF path attribute value of 100 applies at export time only, when the routes are exported from the routing table into BGP.

If a BGP route is received without a LOCAL_PREF attribute, the route is stored in the routing table and advertised by BGP as if it were received with a LOCAL_PREF value of 100. A non-BGP route that is advertised by BGP is advertised with a LOCAL_PREF value of 100 by default.

SEE ALSO

Route Preferences Overview

Example: Configuring the Local Preference Value for BGP Routes

IN THIS SECTION

- Requirements | 297
- Overview | 297
- Configuration | 298
- Verification | 313

This example shows how to configure local preference in internal BGP (IBGP) peer sessions.

Requirements

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

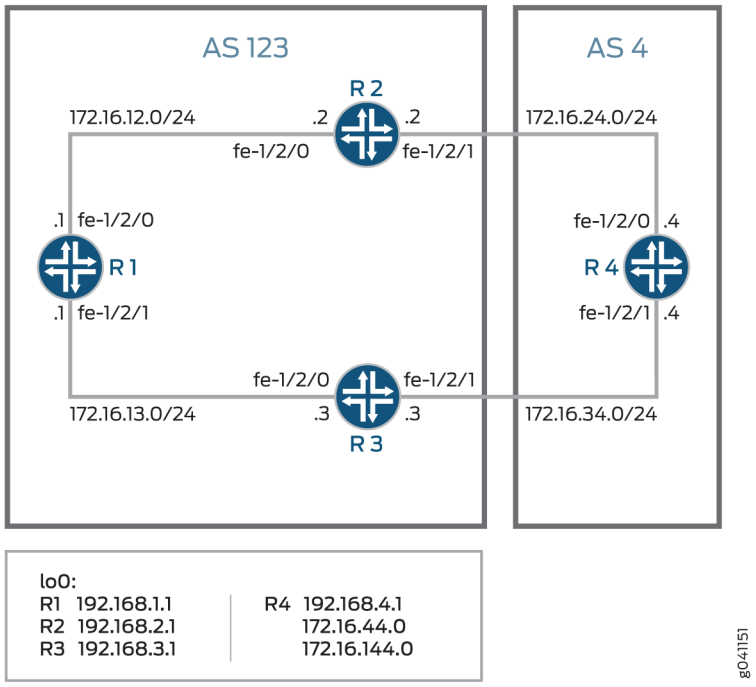
Overview

To change the local preference metric advertised in the path attribute, you must include the `local-preference` statement, specifying a value from 0 through 4,294,967,295 (2^{32} – 1).

There are several reasons you might want to prefer one path over another. For example, compared to other paths, one path might be less expensive to use, might have higher bandwidth, or might be more stable.

Figure 19 on page 298 shows a typical network with internal peer sessions and multiple exit points to a neighboring AS.

Figure 19: Typical Network with IBGP Sessions and Multiple Exit Points



To reach Device R4, Device R1 can take a path through either Device R2 or Device R3. By default, the local preference is 100 for either route. When the local preferences are equal, Junos OS has rules for breaking the tie and choosing a path. (See "Understanding BGP Path Selection" on page 12.) In this example, the active route is through Device R2 because the router ID of Device R2 is lower than the router ID of Device R3. The following example shows how to override the default behavior with an explicit setting for the local preference. The example configures a local preference of 300 on Device R3, thereby making Device R3 the preferred path to reach Device R4.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **299**
- Configuring Device R1 | **301**
- Configuring Device R2 | **304**
- Configuring Device R3 | **307**
- Configuring Device R4 | **310**

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 12.12.12.1/24
set interfaces fe-1/2/1 unit 2 family inet address 13.13.13.1/24
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.1.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.1.1
```

Device R2

```
set interfaces fe-1/2/0 unit 3 family inet address 12.12.12.2/24
set interfaces fe-1/2/1 unit 4 family inet address 24.24.24.2/24
set interfaces lo0 unit 2 family inet address 192.168.2.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.2.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 24.24.24.4
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set policy-options policy-statement send-direct term 1 from protocol direct
```

```
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.2.1
```

Device R3

```
set interfaces fe-1/2/0 unit 5 family inet address 13.13.13.3/24
set interfaces fe-1/2/1 unit 6 family inet address 34.34.34.3/24
set interfaces lo0 unit 3 family inet address 192.168.3.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.3.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 34.34.34.4
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.5
set protocols ospf area 0.0.0.0 interface fe-1/2/1.6
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.3.1
```

Device R4

```
set interfaces fe-1/2/0 unit 7 family inet address 24.24.24.4/24
set interfaces fe-1/2/1 unit 8 family inet address 34.34.34.4/24
set interfaces lo0 unit 4 family inet address 192.168.4.1/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 123
set protocols bgp group external neighbor 34.34.34.3
set protocols bgp group external neighbor 24.24.24.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 4
set routing-options router-id 192.168.4.1
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 1]
user@R1# set family inet address 12.12.12.1/24
[edit interfaces fe-1/2/1 unit 2]
user@R1# set family inet address 13.13.13.1/24
[edit interfaces lo0 unit 1]
user@R1# set family inet address 192.168.1.1/32
```

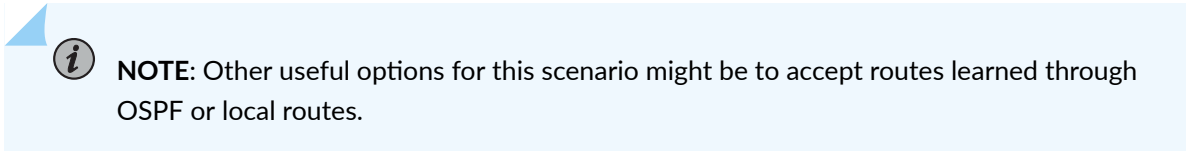
2. Configure BGP.

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set local-address 192.168.1.1
user@R1# set export send-direct
user@R1# set neighbor 192.168.2.1
user@R1# set neighbor 192.168.3.1
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface lo0.1 passive
user@R1# set interface fe-1/2/0.1
user@R1# set interface fe-1/2/1.2
```

4. Configure a policy that accepts direct routes.



NOTE: Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

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

```
[edit routing-options]
user@R1# set autonomous-system 123
user@R1# set router-id 192.168.1.1
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 12.12.12.1/24;
        }
    }
}
fe-1/2/1 {
    unit 2 {
        family inet {
            address 13.13.13.1/24;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.1.1/32;
```

```
        }
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.1.1;
        export send-direct;
        neighbor 192.168.2.1;
        neighbor 192.168.3.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface fe-1/2/0.1;
        interface fe-1/2/1.2;
    }
}
```

```
user@R1# show routing-options
autonomous-system 123;
router-id 192.168.1.1;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 3]
user@R2# set family inet address 12.12.12.21/24
[edit interfaces fe-1/2/1 unit 4]
user@R2# set family inet address 24.24.24.2/24
[edit interfaces lo0 unit 2]
user@R2# set family inet address 192.168.2.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R2# set type internal
user@R2# set local-address 192.168.2.1
user@R2# set export send-direct
user@R2# set neighbor 192.168.1.1
user@R2# set neighbor 192.168.3.1
[edit protocols bgp group external]
user@R2# set type external
user@R2# set export send-direct
user@R2# set peer-as 4
user@R2# set neighbor 24.24.24.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.2 passive
user@R2# set interface fe-1/2/0.3
user@R2# set interface fe-1/2/1.4
```

4. Configure a policy that accepts direct routes.

> **NOTE**: Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

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

```
[edit routing-options]
user@R2# set autonomous-system 123
user@R2# set router-id 192.168.2.1
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 3 {
        family inet {
            address 12.12.12.2/24;
        }
    }
}
fe-1/2/1 {
    unit 4 {
        family inet {
            address 24.24.24.2/24;
        }
    }
}
lo0 {
    unit 2 {
```

```
        family inet {
            address 192.168.2.1/32;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R2# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.2.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.3.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 24.24.24.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface fe-1/2/0.3;
        interface fe-1/2/1.4;
```

```
    }
}
```

```
user@R2# show routing-options
autonomous-system 123;
router-id 192.168.2.1;
```

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

Configuring Device R3

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 Junos OS CLI User Guide.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 5]
user@R3# set family inet address 13.13.13.3/24
[edit interfaces fe-1/2/1 unit 6]
user@R3# set family inet address 34.34.34.3/24
[edit interfaces lo0 unit 3]
user@R3# set family inet address 192.168.3.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R3# set type internal
user@R3# set local-address 192.168.3.1
user@R3# set export send-direct
user@R3# set neighbor 192.168.1.1
user@R3# set neighbor 192.168.2.1
[edit protocols bgp group external]
user@R3# set type external
user@R3# set export send-direct
```

```
user@R3# set peer-as 4
user@R3# set neighbor 34.34.34.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R3# set interface lo0.3 passive
user@R3# set interface fe-1/2/0.5
user@R3# set interface fe-1/2/1.6
```

4. Configure a policy that accepts direct routes.

NOTE: Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R3# set from protocol direct
user@R3# set then accept
```

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

```
[edit routing-options]
user@R3# set autonomous-system 123
user@R3# set router-id 192.168.3.1
```

Results

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

```
user@R3# show interfaces
fe-1/2/0 {
    unit 5 {
        family inet {
            address 13.13.13.3/24;
        }
```

```
    }
}
fe-1/2/1 {
    unit 6 {
        family inet {
            address 34.34.34.3/24;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.3.1/32;
        }
    }
}
```

```
user@R3# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R3# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.3.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.2.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 34.34.34.4;
    }
```

```
}
ospf {
    area 0.0.0.0 {
        interface lo0.3 {
            passive;
        }
        interface fe-1/2/0.5;
        interface fe-1/2/1.6;
    }
}
```

```
user@R3# show routing-options
autonomous-system 123;
router-id 192.168.3.1;
```

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

Configuring Device R4

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 Junos OS CLI User Guide.

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 7]
user@R4# set family inet address 24.24.24.4/24
[edit interfaces fe-1/2/1 unit 8]
user@R4# set family inet address 34.34.34.4/24
[edit interfaces lo0 unit 4]
user@R4# set family inet address 192.168.4.1/32
```

2. Configure BGP.

```
[edit protocols bgp group external]
user@R4# set type external
```

```
user@R4# set export send-direct
user@R4# set peer-as 123
user@R4# set neighbor 34.34.34.3
user@R4# set neighbor 24.24.24.2
```

3. Configure a policy that accepts direct routes.

> **NOTE**: Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R4# set from protocol direct
user@R4# set then accept
```

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

```
[edit routing-options]
user@R4# set autonomous-system 4
user@R4# set router-id 192.168.4.1
```

Results

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

```
user@R4# show interfaces
fe-1/2/0 {
    unit 7 {
        family inet {
            address 24.24.24.4/24;
        }
    }
}
fe-1/2/1 {
    unit 8 {
        family inet {
            address 34.34.34.4/24;
```

```
        }
    }
}
lo0 {
    unit 4 {
        family inet {
            address 192.168.4.1/32;
        }
    }
}
```

```
user@R4# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R4# show protocols
bgp {
    group external {
        type external;
        export send-direct;
        peer-as 123;
        neighbor 34.34.34.3;
        neighbor 24.24.24.2;
    }
}
```

```
user@R4# show routing-options
autonomous-system 4;
router-id 192.168.4.1;
```

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

Verification

IN THIS SECTION

- Checking the Active Path From Device R1 to Device R4 | 313
- Altering the Local Preference to Change the Path Selection | 314
- Rechecking the Active Path From Device R1 to Device R4 | 314

Confirm that the configuration is working properly.

Checking the Active Path From Device R1 to Device R4

Purpose

Verify that the active path from Device R1 to Device R4 goes through Device R2.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 11 destinations, 18 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

12.12.12.0/24       [BGP/170] 00:11:48, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
13.13.13.0/24       [BGP/170] 00:11:48, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
24.24.24.0/24       [BGP/170] 00:11:48, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
34.34.34.0/24       [BGP/170] 00:11:48, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
192.168.2.1/32      [BGP/170] 00:11:48, localpref 100, from 192.168.2.1
                      AS path: I
```

```
                    > to 12.12.12.2 via fe-1/2/0.1
192.168.3.1/32      [BGP/170] 00:11:48, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
192.168.4.1/32     *[BGP/170] 00:05:14, localpref 100, from 192.168.2.1
                      AS path: 4 I
                    > to 12.12.12.2 via fe-1/2/0.1
                    [BGP/170] 00:05:14, localpref 100, from 192.168.3.1
                      AS path: 4 I
                    > to 13.13.13.3 via fe-1/2/1.2
```

Meaning

The asterisk (*) shows that the preferred path is through Device R2. In the default configuration, Device R2 has a lower router ID than Device R3. The router ID is controlling the path selection.

Altering the Local Preference to Change the Path Selection

Purpose

Change the path so that it goes through Device R3.

Action

From configuration mode, enter the `set local-preference 300` command.

```
[edit protocols bgp group internal]
user@R3# set local-preference 300
user@R3# commit
```

Rechecking the Active Path From Device R1 to Device R4

Purpose

Verify that the active path from Device R1 to Device R4 goes through Device R3.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 11 destinations, 17 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

12.12.12.0/24       [BGP/170] 00:16:48, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
13.13.13.0/24       [BGP/170] 00:00:22, localpref 300, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
24.24.24.0/24       [BGP/170] 00:16:48, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
34.34.34.0/24       [BGP/170] 00:00:22, localpref 300, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
192.168.2.1/32      [BGP/170] 00:16:48, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
192.168.3.1/32      [BGP/170] 00:00:22, localpref 300, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
192.168.4.1/32     *[BGP/170] 00:00:21, localpref 300, from 192.168.3.1
                      AS path: 4 I
                    > to 13.13.13.3 via fe-1/2/1.2
```

Meaning

The asterisk (*) shows that the preferred path is through Device R3. In the altered configuration, Device R3 has a higher local preference than Device R2. The local preference is controlling the path selection.

SEE ALSO

BGP Configuration Overview | 24

Example: Configuring BGP to Advertise Inactive Routes

IN THIS SECTION

- Requirements | 317
- Overview | 317
- Configuration | 318
- Verification | 322

By default, BGP readvertises only active routes. To have the routing table export to BGP the best route learned by BGP even if Junos OS did not select it to be an active route, include the `advertise-inactive` statement:

```
advertise-inactive;
```

In Junos OS, BGP advertises BGP routes that are installed or active, which are routes selected as the best based on the BGP path selection rules. The `advertise-inactive` statement allows nonactive BGP routes to be advertised to other peers.

NOTE: If the routing table has two BGP routes where one is active and the other is inactive, the `advertise-inactive` statement does not advertise the inactive BGP prefix. This statement does not advertise an inactive BGP route in the presence of another active BGP route. However, if the active route is a static route, the `advertise-inactive` statement advertises the inactive BGP route.

NOTE: The `advertise-inactive` statement does not help to advertise the inactive route from the VRF when the router is configured as a route reflector.

Junos OS also provides support for configuring a BGP export policy that matches the state of an advertised route. You can match either active or inactive routes, as follows:

```
policy-options {
    policy-statement name{
        from state (active|inactive);
```

```
    }
}
```

This qualifier only matches when used in the context of an export policy. When a route is being advertised by a protocol that can advertise inactive routes (such as BGP), `state inactive` matches routes advertised as a result of the `advertise-inactive` (or `advertise-external`) statement.

For example, the following configuration can be used as a BGP export policy to mark routes advertised due to the `advertise-inactive` setting with a user-defined community. That community can be later used by the receiving routers to filter out such routes from the forwarding table. Such a mechanism can be used to address concerns that advertising paths not used for forwarding by the sender might lead to forwarding loops.

```
user@host# show policy-options
policy-statement mark-inactive {
    term inactive {
        from state inactive;
        then {
            community set comm-inactive;
        }
    }
    term default {
        from protocol bgp;
        then accept;
    }
    then reject;
}
community comm-inactive members 65536:65284;
```

Requirements

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

Overview

IN THIS SECTION

- Topology | 318

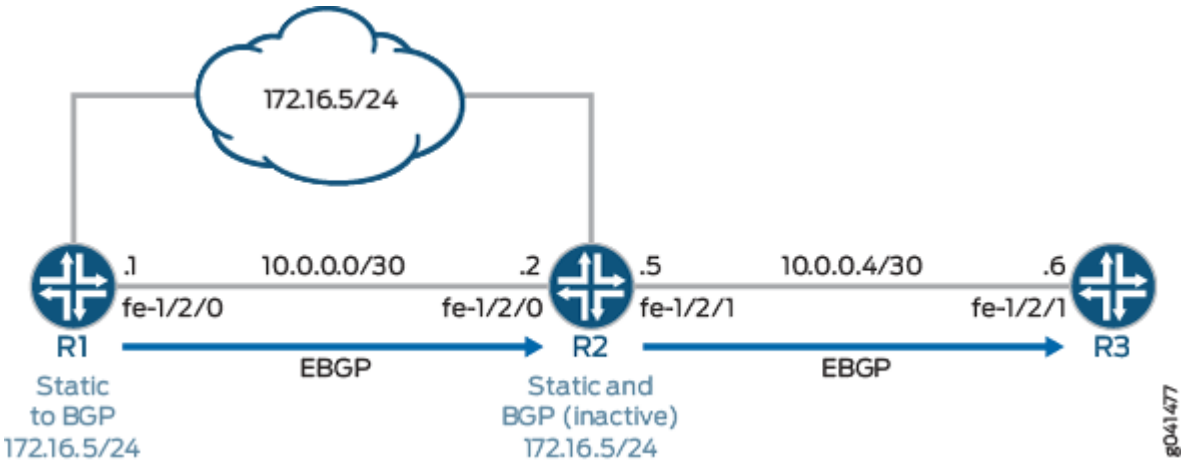
In this example, Device R2 has two external BGP (EBGP) peers, Device R1 and Device R3.

Device R1 has a static route to 172.16.5/24. Likewise, Device R2 also has a static route to 172.16.5/24. Through BGP, Device R1 sends information about its static route to Device R2. Device R2 now has information about 172.16.5/24 from two sources—its own static route and the BGP-learned route received from Device R1. Static routes are preferred over BGP-learned routes, so the BGP route is inactive on Device R2. Normally Device R2 would send the BGP-learned information to Device R3, but Device R2 does not do this because the BGP route is inactive. Device R3, therefore, has no information about 172.16.5/24 unless you enable the `advertise-inactive` command on Device R2, which causes Device R2 to send the BGP-learned to Device R3.

Topology

Figure 20 on page 318 shows the sample network.

Figure 20: BGP Topology for advertise-inactive



"CLI Quick Configuration" on page 319 shows the configuration for all of the devices in Figure 20 on page 318.

The section "No Link Title" on page 320 describes the steps on Device R2.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 319
- Procedure | 320

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 R1

```
set interfaces fe-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 to_R2 type external
set protocols bgp group to_R2 export send-static
set protocols bgp group to_R2 neighbor 10.0.0.2 peer-as 200
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.5.0/24 discard
set routing-options static route 172.16.5.0/24 install
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group to_R1 type external
set protocols bgp group to_R1 neighbor 10.0.0.1 peer-as 100
set protocols bgp group to_R3 type external
set protocols bgp group to_R3 advertise-inactive
set protocols bgp group to_R3 neighbor 10.0.0.6 peer-as 300
set routing-options static route 172.16.5.0/24 discard
set routing-options static route 172.16.5.0/24 install
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.6/30
set interfaces fe-1/2/0 unit 9 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 200
```

```
set protocols bgp group ext neighbor 10.0.0.5
set routing-options autonomous-system 300
```

Procedure

Step-by-Step Procedure

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

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 0 family inet address 10.0.0.5/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group to_R1]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 100
```

3. Configure the EBGP connection to Device R3.

```
[edit protocols bgp group to_R3]
user@R2# set type external
user@R2# set neighbor 10.0.0.6 peer-as 300
```

4. Add the `advertise-inactive` statement to the EBGP group peering session with Device R3.

```
[edit protocols bgp group to_R3]
user@R2# set advertise-inactive
```

5. Configure the static route to the 172.16.5.0/24 network.

```
[edit routing-options static]
user@R2# set route 172.16.5.0/24 discard
user@R2# set route 172.16.5.0/24 install
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.0.0.5/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
```

```
    }
}
```

```
user@R2# show protocols
bgp {
    group to_R1 {
        type external;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
    }
    group to_R3 {
        type external;
        advertise-inactive;
        neighbor 10.0.0.6 {
            peer-as 300;
        }
    }
}
```

```
user@R2# show routing-options
static {
    route 172.16.5.0/24 {
        discard;
        install;
    }
}
autonomous-system 200;
```

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

Verification

IN THIS SECTION

- Verifying the BGP Active Path | 323
- Verifying the External Route Advertisement | 323
- Verifying the Route on Device R3 | 324

- Experimenting with the advertise-inactive Statement | 325

Confirm that the configuration is working properly.

Verifying the BGP Active Path

Purpose

On Device R2, make sure that the 172.16.5.0/24 prefix is in the routing table and has the expected active path.

Action

```
user@R2> show route 172.16.5

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.5.0/24      *[Static/5] 21:24:38
                      Discard
                    [BGP/170] 21:21:41, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
```

Meaning

Device R2 receives the 172.16.5.0/24 route from both Device R1 and from its own statically configured route. The static route is the active path, as designated by the asterisk (*). The static route path has the lowest route preference (5) as compared to the BGP preference (170). Therefore, the static route becomes active.

Verifying the External Route Advertisement

Purpose

On Device R2, make sure that the 172.16.5.0/24 route is advertised toward Device R3.

Action

```
user@R2> show route advertising-protocol bgp 10.0.0.6


inet.0: 6 destinations, 7 routes (6 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.5.0/24           Self                                    100 I
```

Meaning

Device R2 is advertising the 172.16.5.0/24 route toward Device R3

Verifying the Route on Device R3

Purpose

Make sure that the 172.16.6.0/24 prefix is in Device R3's routing table.

Action

```
user@R3> show route 172.16.5.0/24

inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.5.0/24      *[BGP/170] 00:01:19, localpref 100
                      AS path: 200 100 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/1.0
```

Meaning

Device R3 has the BGP-learned route for 172.16.5.0/24.

Experimenting with the advertise-inactive Statement

Purpose

See what happens when the `advertise-inactive` statement is removed from the BGP configuration on Device R2.

Action

1. On Device R2, deactivate the `advertise-inactive` statement.

```
[edit  protocols bgp group to_R3]
user@R2# deactivate advertise-inactive
user@R2# commit
```

2. On Device R2, check to see if the 172.16.5.0/24 route is advertised toward Device R3.

```
user@R2> show route advertising-protocol bgp 10.0.0.6
```

   As expected, the route is no longer advertised.

3. On Device R3, ensure that the 172.16.5/24 route is absent from the routing table.

```
user@R3> show route 172.16.5/24
```

Meaning

Device R1 advertises route 172.16.5/24 to Device R2, but Device R2 has a manually configured static route for this prefix. Static routes are preferred over BGP routes, so Device R2 installs the BGP route as an inactive route. Because the BGP route is not active, Device R2 does not readvertise the BGP route to Device R3. This is the default behavior in Junos OS. If you add the `advertise-inactive` statement to the BGP configuration on Device R2, Device R2 readvertises nonactive routes.

SEE ALSO

Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers | 455

Understanding BGP Path Selection | 12

Change History Table

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

Release	Description
10.4	Starting in Junos OS Release 10.4, if you configure a `static-label-switched-path` the default preference value is 6.

BGP 4-Byte AS Numbers

IN THIS SECTION

- 4-Byte Autonomous System Numbers Overview | 326
- Implementing 4-Byte Autonomous System Numbers | 327
- Configuring 4-Byte Autonomous System Numbers | 329
- Prepending 4-Byte AS Numbers in an AS Path | 330
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333
- Understanding 4-Byte AS Numbers and Route Distinguishers | 337
- Understanding 4-Byte AS Numbers and Route Loop Detection | 338
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341
- Example: Enforcing Correct Autonomous System Number in AS-Path in BGP Network | 343

4-Byte Autonomous System Numbers Overview

This Technology Overview describes 4-byte autonomous system (AS) numbers and the operation of BGP in a network with a mix of 2-byte and 4-byte AS numbers.

The 2-byte AS number, also known as a 16-bit AS number or 2-octet AS number, provides a pool of 65,536 AS numbers. The 2-byte AS number range has been exhausted. 4-byte AS numbers are specified in RFC 4893, *BGP Support for Four-Octet AS Number Space* and provide a pool of 4,294,967,296 AS numbers.

As of January 1, 2009 the Internet Assigned Numbers Authority (IANA) only assigns 4-byte AS numbers, unless a 2-byte AS number is specifically requested. The Internet Engineering Task Force (IETF) RFC 4893 defines a method for smooth transition from 2-byte AS numbers to 4-byte AS numbers and for maintaining backward compatibility.

RFC 4893 introduces two new optional transitive BGP attributes, AS4_PATH and AS4_AGGREGATOR. These new attributes are used to propagate 4-byte AS path information across BGP speakers that do not support 4-byte AS numbers.

RFC 4893 also introduces a reserved, well-known, 2-byte AS number, AS 23456. This reserved AS number is called AS_TRANS in RFC 4893.

RFC 7300, *Reservation of Last Autonomous System (AS) Numbers* and the Internet draft *draft-ietf-idr-as0-06* restrict the use of 2-byte AS number 65535, 4-byte AS number 4294967295UL, and AS number 0 in a configuration. Therefore, when you use these restricted AS numbers, the commit operation fails.

SEE ALSO

- Configuring 4-Byte Autonomous System Numbers | **329**
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | **340**
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | **341**
- Prepending 4-Byte AS Numbers in an AS Path | **330**
- Understanding 4-Byte AS Numbers and Route Distinguishers | **337**
- Understanding 4-Byte AS Numbers and Route Loop Detection | **338**
- Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | **333**

Implementing 4-Byte Autonomous System Numbers

Junos OS Release 9.1 and later supports 4-byte AS numbers.

If your network is currently using 2-byte AS numbers, you are not required to get new 4-byte AS numbers. The 2-byte AS number range is a subset of the 4-byte AS number range. A Juniper networks

router that supports 4-byte AS numbers simply prepends a string of zeros in front of the 2-byte AS number. For example, the 2-byte AS number 65000 becomes the 4-byte AS number 00000.65000.

If your Juniper Networks router supports 4-byte AS numbers and has a peer relationship with a router that does not support 4-byte AS numbers, the following sequence takes place in the adjacent RIB-in routing table after the router that supports 4-byte AS numbers advertises this capability to the new peer:

1. The router that supports 4-byte AS numbers receives an advertisement from the peer that supports only 2-byte AS numbers.
2. On the router that supports 4-byte AS numbers, the 2-byte AS path is converted into the 4-byte AS number by prepending a string of zeros in front of the 2-byte AS number.
3. If a 4-byte AS number is also present in the path, it is merged with the 2-byte AS numbers in the path.
4. If the AGGREGATOR and AS4_AGGREGATOR attributes are present, these attributes are also merged.

If your Juniper Networks router supports 4-byte AS numbers and has a peer relationship with a router that does not support 4-byte AS numbers, the following sequence takes place in the adjacent RIB-out routing table:

1. Update message are reformatted before being sent to the router that does not support 4-byte AS numbers.
2. The router that supports 4-byte AS numbers sends the 4-byte AS number in the AS4_PATH attribute.
3. The AS_PATH attribute is also sent. It is encoded with the 2-byte AS numbers. Mappable 4-byte AS numbers, below 64537, are sent as 2-byte AS numbers. Non-mappable 4-byte AS numbers, above 64536, are represented by the well-known 2-byte AS number, AS 23456.
4. A single peer group is used for the routers that support 4-byte AS numbers and the routers that support only 2-byte AS numbers.

SEE ALSO

- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341

Prepending 4-Byte AS Numbers in an AS Path | 330
Understanding 4-Byte AS Numbers and Route Distinguishers | 337
Understanding 4-Byte AS Numbers and Route Loop Detection | 338
Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Configuring 4-Byte Autonomous System Numbers

This section describes how to configure a 4-byte AS number and how to verify if the BGP peer supports 4-byte AS numbers.

The AS number can be specified in plain number format or in AS-dot notation format on routers running Junos OS Release 9.2 and later. For example, the 4-byte AS number of 65,546 is represented in plain-number format as 65546. The same AS number is represented in AS-dot notation format as 1.10 on routers running Junos OS Release 9.2 and later.

- To configure a 4-byte AS number in AS-dot notation format, include the `autonomous-system` statement and specify the 4-byte AS number. In the following example the AS number is set to `1.10`.

```
user@host# set routing-options autonomous-system 1.10
```

- To configure a 4-byte AS number in plain number format, include the `autonomous-system` statement and specify the 4-byte AS number. In the following example the AS number is set to `65546`.

```
user@host# set routing-options autonomous-system 65546
```

- After a BGP peer session has been negotiated, you can verify whether the peer supports 4-byte AS numbers or not. To verify whether the peer supports 4-byte AS numbers or not, use the `show bgp neighbor` command. In the following example the peer does not support 4-byte AS numbers.

```
user@host# show bgp neighbor 192.168.1.9 | match "AS"
Peer: 192.168.1.9+179 AS 65056 Local: 192.168.1.3+52616 AS 65000
  Peer does not support 4 byte AS extension
```

- In the following example the peer does support 4-byte AS numbers.

```
user@host# show bgp neighbor 192.168.1.9 | match "AS"
Peer: 192.168.1.10+52679 AS 1000000000 Local: 192.168.1.3+179 AS 65000
  Peer supports 4 byte AS extension (peer-as 1000000000)
```

SEE ALSO

4-Byte Autonomous System Numbers Overview | 326

Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341

Implementing 4-Byte Autonomous System Numbers | 327

Understanding 4-Byte AS Numbers and Route Distinguishers | 337

Understanding 4-Byte AS Numbers and Route Loop Detection | 338

Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection | 254

Prepending 4-Byte AS Numbers in an AS Path

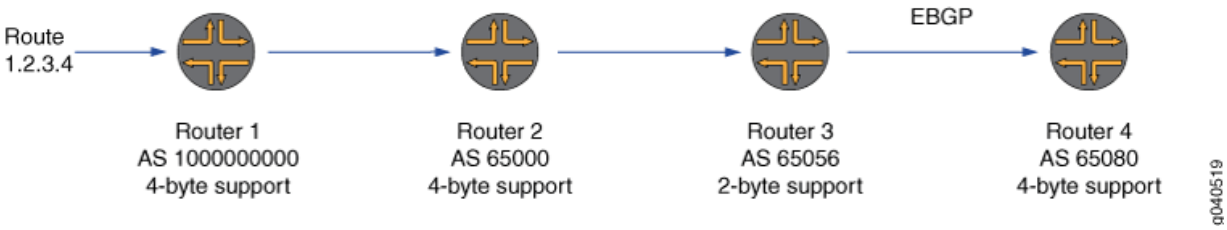
When an address prefix advertisement transits a domain, the domain effectively “signs” the prefix advertisement by prepending its autonomous system number (ASN) to the AS path associated with the address prefix. At any point in the network the AS path describes a sequence of connected domains that forms a path from the current point to the originating domain. The left-most number in the AS path list is the ASN of the adjacent AS from which the address prefix advertisement was received. The sequence of numbers indicates the sequence of ASs though which this update was propagated.

This section describes how to prepend one or more AS numbers at the beginning of an AS path. The AS numbers are added at the beginning of the path after the actual AS number from which the route originates has been added to the path. Prepending an AS path makes a shorter AS path look longer and therefore less preferable to BGP.

NOTE: As of Junos OS Release 15.1, the `enforce-first-as` statement enforces the first (left-most) autonomous system number (ASN) in AS-path is the previous neighbor's ASN as the domain is transited.

In Figure 21 on page 331, Router 2 is configured to prepend AS 1000000000 4 times in front of AS number 65000.

Figure 21: EBGP with 4-Byte AS Numbers Prepended to the AS Path



You can display the route details using the show route command on Router 3. In the following example, notice that the prepended AS number displayed in the AS path on Router 3 is the AS_TRANS number, AS 23456. This is because Router 3 does not support 4-byte AS numbers.

```
user@Router3# show route 1.2.3.4 detail
...
1.2.3.4/32        *[BGP/170] 01:39:55, localpref 100, from 192.168.1.3
                      AS path: 65000 23456 23456 23456 23456 I
```

You can display the route details using the show route command on Router 4. In the following example, notice that the prepended AS number displayed in the AS path on Router 4 is AS 1000000000. This is because Router 4 supports 4-byte AS numbers and merges the AS_PATH and AS4_PATH attributes.

```
user@Router4# show route 1.2.3.4 detail
...
1.2.3.4/32        *[BGP/170] 01:39:55, localpref 100, from 192.168.1.9
                      AS path: 65056 65000 1000000000 1000000000 1000000000 1000000000 I
```

SEE ALSO

enforce-first-as

4-Byte Autonomous System Numbers Overview | 326

Configuring 4-Byte Autonomous System Numbers | 329

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341

Implementing 4-Byte Autonomous System Numbers | 327

Understanding 4-Byte AS Numbers and Route Distinguishers | 337

Understanding 4-Byte AS Numbers and Route Loop Detection | 338

Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Configuring 4-Byte AS Numbers and BGP Extended Community Attributes

A BGP community is a group of destinations that share a common property. You can configure the standard community attribute and extended community attributes for inclusion in BGP update messages.

For example, when configuring a VPN routing and forwarding (VRF) instance, you need to configure a route target. A route target is one type of BGP extended community attribute. To create a named BGP extended community attribute, include the `community` statement and specify the community members:

```
community name {
    members [ community-ids ];
}
```

To specify the community members, you must specify the community ID. The community ID consists of three components that you specify in the following format:

```
type:administrator:assigned-number
```

The *administrator* field of some BGP extended community attributes is an AS number. To configure a `target` extended community, which includes a 4-byte AS number in the plain-number format, append the letter “L” to the end of the number.

In the following example, a target community with the 4-byte AS number 334324 and an assigned number of 132 is represented as target:334324L:132.

```
[edit policy-options]
community vpn_blue members [ target:334324L:132 ];
```

NOTE: If you display the target extended community information on a peer router that does not support 4-byte AS numbers, the router displays target:unknown format.

SEE ALSO

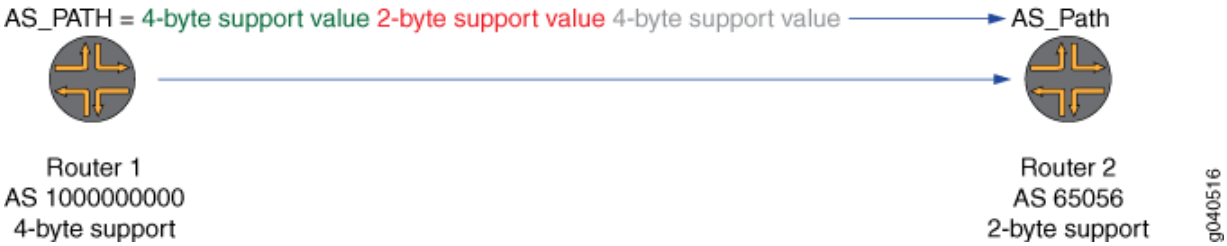
- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte Autonomous System Numbers | 329
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341
- Implementing 4-Byte Autonomous System Numbers | 327
- Prepending 4-Byte AS Numbers in an AS Path | 330
- Understanding 4-Byte AS Numbers and Route Distinguishers | 337
- Understanding 4-Byte AS Numbers and Route Loop Detection | 338

Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain

This section describes what happens when a router that supports 4-byte AS numbers sends the AS path statement to a router that only supports 2-byte AS numbers if the first router is configured with an AS number outside the 2-byte AS number range.

In Figure 22 on page 334 Router 1 supports 4-byte AS numbers. Router 1 is configured to use a 4-byte AS number, AS 1000000000. Router 2 supports 2-byte AS numbers. Router 2 is configured with a 2-byte AS number, AS 65056.

Figure 22: 4-Byte Capable Router AS Path to a 2-Byte Capable Router

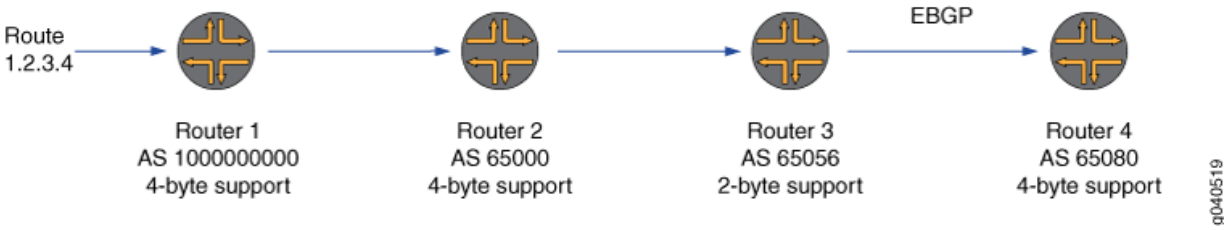


- Router 2 does not accept 4-byte AS numbers in the AS_PATH attribute. You can verify this using the `show bgp neighbor` command on Router 1.

```
user@Router1# show bgp neighbor 192.168.1.9 | match "AS"
Peer: 192.168.1.9+179 AS 65056 Local: 192.168.1.2+64053 AS 65080
  Peer does not support 4 byte AS extension
```

Figure 23 on page 334 shows four routers running EBGP. Router 1, Router 2, and Router 4 support 4-byte AS numbers. Router 3 does not support 4-byte AS numbers.

Figure 23: EBGP 4-Byte AS Path Through a 2-Byte AS Domain



In this case:

- Router 1 sends the 4-byte AS number, AS 1000000000, in the AS_PATH attribute to Router 2.
- Router 2 knows that Router 3 does not support 4-byte AS numbers.
- Router 2 sends the AS_TRANS number, AS 23456, in the AS_PATH attribute in place of the 4-byte AS number to Router 3.
- Router 2 sends the 4-byte AS number, AS 1000000000 in the AS4_PATH attribute to Router 3.
- Because the AS4_PATH attribute is transitive, Router 3 sends both the AS_PATH attribute and the AS4_PATH attribute to Router 4.

- When Router 4 receives the AS_PATH and AS4_PATH attributes, it merges the path statements to create an accurate AS path.

You can display the AS path using the `show route` command on Router 3. In the following example, notice that the AS number 23456 appears in the AS path and that the AS4_PATH attribute is `Unrecognized`. Because the AS4_PATH attribute is a transitive attribute, it is forwarded to the next router.

```
user@Router3# show route 1.2.3.4 detail
AS path: 65000 23456 I Unrecognized Attributes: 13 bytes
```

You can display the route details using the `show route` command on Router 4. In the following example, notice that as the AS path transitions Router 3, as shown in the AS2 (2-byte AS) path, the AS number is displayed as AS_TRANS. This means that Router 3 sees the AS number as 23456. In the AS4 (4-byte AS) path the AS number is displayed as 1000000000. In the merged AS path the correct AS path numbers are displayed for AS 65056, AS 65000, and AS 1000000000.

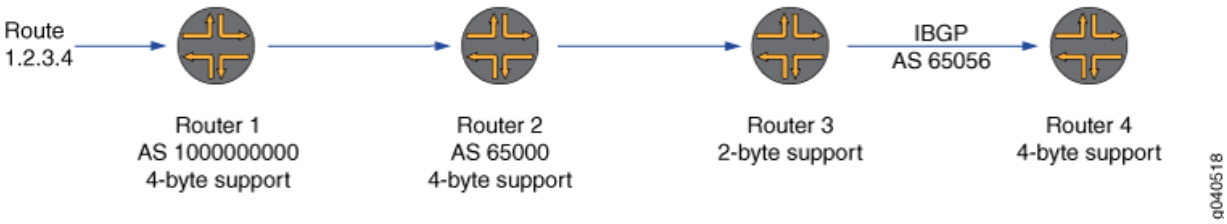
```
user@Router4# show route 1.2.3.4 detail
...
AS path: AS2 PA[3]:65056 65000 AS_TRANS

AS path: AS4 PA[2]:65056 1000000000

AS path: Merged[3]:65056 65000 1000000000 I
```

Figure 24 on page 335 shows 4 routers running IBGP. Router 1, Router 2, and Router 4 support 4-byte AS numbers. Router 3 does not support 4-byte AS numbers.

Figure 24: IBGP 4-Byte AS Path Through a 2-Byte AS Domain



In this case:

- Router 1 sends the 4-byte AS number, AS 1000000000, in the AS_PATH attribute to Router 2.
- Router 2 knows that Router 3 does not support 4-byte AS numbers.

- Router 2 sends the AS_TRANS number, AS 23456, in the AS_PATH attribute in place of the 4-byte AS number to Router 3.
- Router 3 sends both the AS_PATH attribute and the AS4_PATH attribute to Router 4.
- When Router 4 receives the AS_PATH and AS4_PATH attributes, it merges the path statements to create an accurate AS path.

You can display the route details using the `show route` command on Router 2. In the following example, notice that the AS path is displayed as 1000000000.

```
user@Router2# show route 1.2.3.4 detail
...
AS path: 1000000000
```

You can display the route details using the `show route` command on Router 3. In the following example, notice that the AS path is displayed as 65000 23456.

```
user@Router3# show route 1.2.3.4 detail
...
AS path: 65000 23456 I
```

You can display the route details using the `show route` command on Router 4. In the following example, notice that the merged AS path is displayed as 65000 1000000000.

```
user@Router4# show route 1.2.3.4 detail
...
AS path: 65000 1000000000 I
```

SEE ALSO

- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Configuring 4-Byte Autonomous System Numbers | 329
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341
- Implementing 4-Byte Autonomous System Numbers | 327

Prepending 4-Byte AS Numbers in an AS Path | 330

Understanding 4-Byte AS Numbers and Route Loop Detection | 338

Understanding 4-Byte AS Numbers and Route Distinguishers

A route distinguisher (RD) is an 8-byte field prefixed to a service provider customer's IPv4 address. The resulting 12-byte field is a unique VPN-IPv4 address. The RD in BGP messages consists of two major fields, the type field (2 bytes) and value field (6 bytes). The type field determines how the value field should be interpreted.

The route distinguisher is configured as a 6-byte value that you can specify as *as-number*:*number*, where *as-number* is your assigned AS number and *number* (also known as an administrative number or assigned number subfield) is any 2-byte or 4-byte value. The AS number can be in the range from 1 through 4,294,967,295. If the AS number is a 2-byte value, the administrative number is a 4-byte value. If the AS number is 4-byte value, the administrative number is a 2-byte value.

An RD consisting of a 4-byte AS number and a 2-byte administrative number is defined as a type 2 route distinguisher in RFC 4364, *BGP/MPLS IP Virtual Private Networks*.

To configure an RD using a 4-byte AS number, append the letter “L” to the end of the number. In the following example, the 4-byte AS number is 7765000 and the administrative number is 1000:

```
user@Router1# set routing-instances 4B route-distinguisher 7765000L:1000
```

If the router you are configuring is a BGP peer of a router that does not support 4-byte AS numbers, you also need to configure a local AS number as discussed in "Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number" on page 341. To configure the local AS number, include the `local-as` statement, specify the 2-byte AS number to use (65001), and include the `private` option.

```
user@Router1# set routing-instances 4B protocols bgp group 4B2Bpeers local-as 65001 private
```

SEE ALSO

4-Byte Autonomous System Numbers Overview | 326

Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332

Configuring 4-Byte Autonomous System Numbers | 329

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number | 341

Implementing 4-Byte Autonomous System Numbers | 327

Prepending 4-Byte AS Numbers in an AS Path | 330

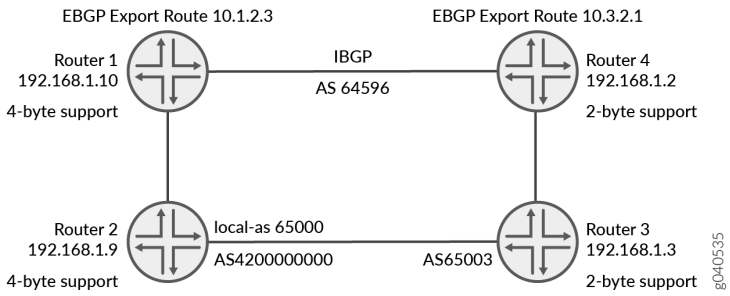
Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Understanding 4-Byte AS Numbers and Route Loop Detection

One of the most important functions in BGP is route loop detection at the autonomous system level using the AS_PATH attribute. A simple way of thinking of the AS_PATH is that it is the list of autonomous systems that a route goes through to reach its destination. Loops are detected and avoided by the router checking for its own AS number in the AS_PATH received from a neighboring AS.

This section describes how route loop detection works with a mix of routers that support and do not support 4-byte AS numbers. Figure 25 on page 338 shows a small network with the potential for BGP loops.

Figure 25: 4-Byte AS Numbers and Loop Detection



In the first example, an EBGP route, route 10.1.2.3, is first advertised by Router 1. The first AS in the path is AS 64596 as configured on Router 1. The second AS that is in the path is AS 4200000000 as configured on Router 2. AS 4200000000 is sent in the AS4_path attribute and the AS_TRANS number, AS 23456, is sent in the AS_PATH attribute to Router 3. The third AS in the path is AS 65003, as configured on Router 3.

The `show route` command output shows the AS path for route 10.1.2.3 as advertised by Router 3 to Router 4. In the `show route` command output, you see AS 64596 first. Because Router 3 does not support 4-byte AS numbers, you see AS 23456 second. Because Router 2 used a local AS of 65000 to establish a

peer relationship with Router 3, you see AS 65000 third. AS 65003 is not in the `show route` command output because the command was entered on the router configured with AS 65003.

```
user@Router3# show route advertising-protocol bgp 192.168.1.2
...
Prefix Nexthop MED Lclpref AS path
10.2.3.4/32 Self   65000 23456 64596 I
```

In this case, when Router 4 sees its own AS number, AS 64596, in the path, it detects a routing loop.

In the second example, an EBGP route, route 10.3.2.1, is first advertised by Router 4. The first AS in the path is AS 60596 as configured on Router 4. The second AS in the path is AS 65003 as configured on Router 3. The third AS is in the path is AS 4200000000 as configured on Router 2.

The `show route` command output shows the AS path for route 10.3.2.1 as advertised by Router 2 to Router 1. In the `show route` command output, you see AS 64596 first and AS 65003 second. AS 4200000000 is not in the `show route` command output because the command was entered on the router configured with AS 4200000000.

```
user@Router2# show route advertising-protocol bgp 192.168.1.10
...
Prefix Nexthop MED Lclpref AS path
10.3.2.1/32 Self   65003 64596 I
```

When Router 1 sees its own AS number, AS 64596, in the path, it detects a routing loop.

SEE ALSO

- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Configuring 4-Byte Autonomous System Numbers | 329
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Implementing 4-Byte Autonomous System Numbers | 327
- Prepending 4-Byte AS Numbers in an AS Path | 330
- Understanding 4-Byte AS Numbers and Route Distinguishers | 337
- Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333
- Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection | 254

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number

This section describes what happens when a router that supports 4-byte AS numbers establishes a peer relationship with a router that only supports 2-byte AS numbers if both routers are configured with AS numbers in the 2-byte AS number range.

In Figure 26 on page 340, Router 1 is running Junos OS Release 9.2 that supports 4-byte AS numbers. Router 1 is configured to use a 2-byte AS number, AS 12596. Router 2 is running Junos OS Release 8.5 that supports 2-byte AS numbers. Router 2 is configured with a 2-byte AS number, AS 60000.

Figure 26: 4-Byte Capable Router Having a Peer Relationship with a 2-Byte Capable Router Using a 2-Byte AS Number

- The following example shows the relevant portion of the Router 1 configuration.

```
user@Router1# show configuration
…
autonomous-system 12596;
…
local-address 192.168.1.10;
export static-to-bgp;
peer-as 60000;
```

- To verify that the AS path of route 1.2.3.4 contains AS 12596, use the `show route` command on Router 2. The following example shows that the BGP peer session is established in the normal way and that the AS path of route 1.2.3.4 contains AS 12596:

```
user@Router2# show route 1.2.3.4
1.2.3.4/32         *[BGP/170] 00:01:29, localpref 100, from 192.168.1.10
                      AS path: 12596 I
                    > via at-0/1/0.1001
```

- To display the session-establishment messages logged on Router 1, use the `show log messages` command. The following example shows that Router 1 discovers that Router 2 does not support 4-byte AS numbers:

```
user@Router1# show log messages
Nov  7 09:41:39.443493 bgp_4byte_aspath_add_cap():153 AS4-Peer 192.168.1.9 (External AS 60000)
(SEND): 4 byte AS capability added, AS 12596
Nov  7 09:41:39.443582 bgp_send: sending 67 bytes to 192.168.1.9 (External AS 60000)
[…]
Nov  7 09:41:39.448055 bgp_4byte_aspath_adjust():1279 AS4-Peer 192.168.1.9 (External AS 60000)
(SEND): Adjust BGP update to Old/New BGP speaker format
Nov  7 09:41:39.448132 bgp_4byte_aspath_adjust():1290 AS4-Peer 192.168.1.9 (External AS 60000)
(SEND): Cached information of previous update format is not used
Nov  7 09:41:39.448162 bgp_generate_2byte_aspath():422 AS4-Peer 192.168.1.9 (External AS
60000)(SEND): Generating 2 byte AS path from 4 byte as-path
Nov  7 09:41:39.448198 bgp_send: sending 64 bytes to 192.168.1.9 (External AS 60000)
```

SEE ALSO

- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Configuring 4-Byte Autonomous System Numbers | 329
- Implementing 4-Byte Autonomous System Numbers | 327
- Prepending 4-Byte AS Numbers in an AS Path | 330
- Understanding 4-Byte AS Numbers and Route Distinguishers | 337
- Understanding 4-Byte AS Numbers and Route Loop Detection | 338
- Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 4-Byte AS Number

This section describes what happens when a router that supports 4-byte AS numbers establishes a peer relationship with a router that only supports 2-byte AS numbers if the first router is configured with an AS number outside the 2-byte AS number range.

In Figure 27 on page 342, Router 2 is running Junos OS Release 9.2 that supports 4-byte AS numbers. Router 2 is configured to use a 4-byte AS number, AS 1000000. Router 3 is running Junos OS Release 8.5 that supports 2-byte AS numbers. Router 3 is configured with a 2-byte AS number, AS 60000.

Figure 27: 4-Byte Capable Router Having a Peer Relationship with a 2-Byte Capable Router Using a 4-Byte AS Number

You can configure a local AS number to be used only during the establishment of the BGP session with a BGP neighbor, but to be hidden in the AS path sent to external BGP peers. To configure the local AS number, include the `local-as` statement, specify the 2-byte AS number to use, 65530, and include the `private` option. With this configuration, only the global AS number, 1000000, is included in the AS path sent to external peers. The following example shows the relevant portion of the Router 2 configuration:

```
user@Router2# show configuration
…
autonomous-system 1000000;
…
local-address 192.168.1.9;
export static-to-bgp;
neighbor 192.168.1.3 {
    peer-as 60000;
    local-as 65530 private;
}
```

The peer AS number on Router 3 should equal the local AS number on Router 1. The following example shows the relevant portion of the Router 3 configuration:

```
user@Router3# show configuration
…
autonomous-system 60000;
…
local-address 192.168.1.3;
```

```
neighbor 192.168.1.9 {
    peer-as 65530;
```

To verify that the AS path of route 22.1.2.3 contains AS 65530, use the `show route` command on Router 3. The following example shows that the BGP peer session is established and that the AS path of route 22.1.2.3 contains AS 65530:

```
user@Router3# show route 22.1.2.3
...
22.1.2.3/32        *[BGP/170] 01:39:55, localpref 100, from 192.168.1.9
                      AS path: 65530 I
                    > via so-1/0/3.0
```

SEE ALSO

- 4-Byte Autonomous System Numbers Overview | 326
- Configuring 4-Byte AS Numbers and BGP Extended Community Attributes | 332
- Configuring 4-Byte Autonomous System Numbers | 329
- Establishing a Peer Relationship Between a 4-Byte Capable Router and a 2-Byte Capable Router Using a 2-Byte AS Number | 340
- Implementing 4-Byte Autonomous System Numbers | 327
- Prepending 4-Byte AS Numbers in an AS Path | 330
- Understanding 4-Byte AS Numbers and Route Distinguishers | 337
- Understanding 4-Byte AS Numbers and Route Loop Detection | 338
- Understanding a 4-Byte Capable Router AS Path Through a 2-Byte Capable Domain | 333

Example: Enforcing Correct Autonomous System Number in AS-Path in BGP Network

IN THIS SECTION

- Requirements | 344
- Overview | 344

- Configure enforce-first-as Statement to Check Routes | 345
- Verification | 348

This example shows how the `enforce-first-as` statement, set at the `[edit protocols bgp]` hierarchy level, can be used as a security measure. Configuring this statement creates a consistency check to ensure a BGP peer is a legitimate sender of routing information.

Requirements

Before you begin, set up an BGP network of at least three autonomous systems. Three separate routers is sufficient.

Overview

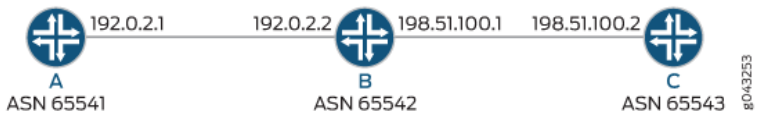
IN THIS SECTION

- Topology | 344

The `enforce-first-as` statement enforces that the first (left-most) autonomous system number (ASN) in the AS-path is consistent with the advertising neighbor's ASN.

The topology is set up with Router C advertising in BGP a static route to Router B, which then readvertises the route to Router A. Then an export policy towards Router A to prepend an unrelated ASN is added to Router B. Lastly, the `enforce-first-as` statement is configured on Router A towards Router B. When Router A gets AS-path, it checks if the left-most ASN in the AS-path is the previous neighbor's ASN and invalidates the route coming from Router B.

Topology



Configure enforce-first-as Statement to Check Routes

IN THIS SECTION

- CLI Quick Configuration | 345
- Procedure | 346

CLI Quick Configuration

To quickly configure the initial configuration for 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.

Initial Configuration on Router A

```
set interfaces ge-1/0/0 unit 0 family inet address 192.0.2.1/29
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.1/32
set routing-options router-id 10.127.0.1
set routing-options autonomous-system 65541
set protocols mpls interface ge-1/0/0.0
set protocols bgp group pe type external
set protocols bgp group pe peer-as 65542
set protocols bgp group pe neighbor 192.0.2.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ldp interface ge-1/0/0.0
set protocols ldp interface lo0.0
```

Initial Configuration on Router B

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/29
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.1/29
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.2/32
set routing-options router-id 10.127.0.2
set routing-options autonomous-system 65542
```

```
set protocols bgp group pe1 type external
set protocols bgp group pe1 peer-as 65541
set protocols bgp group pe1 neighbor 192.0.2.1
set protocols bgp group pe3 type external
set protocols bgp group pe3 peer-as 65543
set protocols bgp group pe3 neighbor 198.51.100.2
```

Initial Configuration on Router C

```
set interfaces ge-1/0/0 unit 0 family inet address 198.51.100.2/29
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.3/32
set routing-options router-id 10.127.0.3
set routing-options autonomous-system 65543
set protocols mpls interface ge-1/0/0.0
set protocols bgp group pe type external
set protocols bgp group pe peer-as 65542
set protocols bgp group pe neighbor 198.51.100.1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0
set protocols ldp interface ge-1/0/0.0
set protocols ldp interface lo0.0
```

Procedure

Step-by-Step Procedure

1. Configure a static route on Router C.

```
C-re0# set routing-options static route 198.51.100.17/29 next-hop 198.51.100.20
C-re0# set routing-options static route 198.51.100.17/29 readvertise
C-re0# commit
```

2. Configure an export policy for the static route.

```
C-re0# set policy-options policy-statement export-static from protocol bgp
C-re0# set policy-options policy-statement export-static then accept
C-re0# set protocols bgp group pe export export-static
C-re0# commit
```

3. Verify that the static route is getting through to Router B and Router A.

```
B-re0# run show route 198.51.100.17
inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.17/29      *[BGP/170] 00:11:40, localpref 100
                      AS path: 65543 I, validation-state: unverified
                    > to 198.51.100.2 via ge-0/0/1.0

A-re0# run show route 198.51.100.17
inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.17/29      *[BGP/170] 00:10:31, localpref 100
                      AS path: 65542 65543 I, validation-state: unverified
                    > to 192.0.2.2 via ge-1/0/0.0
```

Notice that on Router A, route is shown with an AS-path of 65542 65543. Route from Router B to Router A has had the ASN for Router A prepended to the AS-path.

4. Set an export policy to prepend ASN from Router B.

```
B-re0# set policy-options policy-statement as-prepender from neighbor 198.51.100.2
B-re0# set policy-options policy-statement as-prepender then as-path-prepend 65555
B-re0# set protocols bgp group pe1 export as-prepender
B-re0# commit
```

5. Verify route 198.51.100.17 on Router A.

```
A-re0# run show route 198.51.100.17
inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.17/29      *[BGP/170] 00:00:50, localpref 100
                      AS path: 65555 65542 65543 I, validation-state: unverified
                    > to 192.0.2.2 via ge-1/0/0.0

[edit]
A-re0#
```

Notice that ASN 65555 is prepended to the AS path.

6. Configure the `enforce-first-as` statement on Router A.

```
A-re0# set protocols bgp enforce-first-as
A-re0# commit
```

When you check the route again, you see that route 198.51.100.17 is no longer getting through on Router A.

Verification

IN THIS SECTION

- Verify the BGP Session | **348**
- Verify the Static Route | **349**
- Verify Prepend Export Policy | **350**
- Verify the enforce-first-as Statement Is Working | **351**

Verify the BGP Session

Purpose

Verify that a BGP session has been established and with which neighbors the router has established a peering session with.

Action

From operational mode, run the `show bgp summary` command.

```
B-re0> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.0.2.1                65541        367        369       0       0     2:43:57
```

```
0/0/0/0              0/0/0/0
198.51.100.2                65543        369        368       0       0     2:44:00
0/0/0/0              0/0/0/0
```

Meaning

The first line shows the number of groups configured and the number of peers that are up or down. This output shows there are two peers, 192.0.2.1 and 198.51.100.2, up. The table portion shows that there are no paths in the inet.0 table. We can see that Router B has two peers, 65541 and 65543. When the `State` column shows three numbers separated by slashes, the BGP session is up.

Verify the Static Route

Purpose

Verify that a static route is being exported to routers B and A from Router C.

Action

From operational mode, run the `show bgp neighbor` command.

```
C-re0#> show bgp neighbor
Peer: 198.51.100.1+179 AS 65542       Local: 198.51.100.2+62588 AS 65543
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ export-static ]
```

From operational mode, run the `show bgp summary` command.

```
B-re0> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.0.2.1                 65541          8         10       0       0        2:59
0/0/0/0              0/0/0/0
```

```
198.51.100.2                65543         10         10       0       0        3:02
1/1/1/0             0/0/0/0
```

From operational mode, run the `show route protocol bgp` command.

```
A-re0> show route protocol bgp
inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.17/29      *[BGP/170] 00:12:35, localpref 100
                       AS path: 65542 65543 I, validation-state: unverified
                     > to 192.0.2.2 via ge-1/0/0.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

Meaning

With the `show bgp neighbor` command you can see the export policy by name.

With the `show bgp summary` command you can see that there is now one route in the inet.0 table, showing that the table has learned this route.

The `show route protocol bgp` command confirms that the router is learning routes. You can see the route and the AS path. Notice that in Router A we can see the AS path is appended with the ASNs of Routers C and B (65543 and 65542).

Verify Prepend Export Policy

Purpose

Verify ASNs are in AS path of router receiving from Router B.

show bgp neighbor. Lists the BGP routers to which this router is connected. Shows which neighbors the router has established peering sessions with.

show bgp summary. Lists BGP group, peer, and session state information. Helps determine whether a BGP session has been established.

show route protocol bgp. Lists the routes learned from BGP. Confirms that the router is learning routes only from desired neighbors.

Action

From operational mode, run the `show route protocol bgp` command.

```
A-re0> show route protocol bgp
inet.0: 49 destinations, 49 routes (49 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.17/29      *[BGP/170] 00:00:24, localpref 100
                       AS path: 65555 65542 65543 I, validation-state: unverified
                     > to 192.0.2.2 via ge-1/0/0.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

Meaning

You can see that 65555 has been prepended to the AS path.

Verify the enforce-first-as Statement Is Working

Purpose

Verify that the router is learning routes only from desired neighbors.

Action

Verify route 198.51.100.17.

```
A-re0> show route 198.51.100.17 all detail
inet.0: 49 destinations, 49 routes (48 active, 0 holddown, 1 hidden)
198.51.100.17/29 (1 entry, 0 announced)
         BGP                /-101
                Next hop type: Router, Next hop index: 581
```

```
                Address: 0x9db5ad0
                Next-hop reference count: 1
                Source: 192.0.2.2
                Next hop: 192.0.2.2 via ge-1/0/0.0, selected
                Session Id: 0x141
                State: <Hidden Ext>
                Local AS:   65541 Peer AS:   65542
                Age: 1w2d 23:48:47
                Validation State: unverified
                Task: BGP_65542.192.0.2.2
                AS path: 65555 65542 65543 I
                Localpref: 100
                Router ID: 10.127.0.2
                Hidden reason: fails enforce-first-as check
```

If you issue the `show route` command, the route information is not displayed.

```
A-re0> show route 198.51.100.17

A-re0>
```

Meaning

The static route is hidden because it contained an unrelated ASN and the `enforce-first-as` statement was configured.

SEE ALSO

enforce-first-as

Change History Table

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

Release	Description
9.1	Junos OS Release 9.1 and later supports 4-byte AS numbers.

BGP MED Attribute

IN THIS SECTION

- Understanding the MED Attribute That Determines the Exit Point in an AS | 353
- Example: Configuring the MED Attribute That Determines the Exit Point in an AS | 356
- Example: Configuring the MED Using Route Filters | 376
- Example: Configuring the MED Using Communities | 396
- Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates | 397

Understanding the MED Attribute That Determines the Exit Point in an AS

The BGP multiple exit discriminator (MED, or MULTI_EXIT_DISC) is a non-transitive attribute, meaning that it is not propagated throughout the Internet, but only to adjacent autonomous systems (ASs). The MED attribute is optional, meaning that it is not always sent with the BGP updates. The purpose of MED is to influence how other ASs enter your AS to reach a certain prefix.

The MED attribute has a value that is referred to as a *metric*. If all other factors in determining an exit point are equal, the exit point with the lowest metric is preferred.

If a MED is received over an external BGP link, it is propagated over internal links to other BGP-enabled devices within the AS.

BGP update messages include a MED metric if the route was learned from BGP and already had a MED metric associated with it, or if you configure the MED metric in the configuration file.

A MED metric is advertised with a route according to the following general rules:

- A more specific metric overrides a less specific metric. That is, a group-specific metric overrides a global BGP metric, and a peer-specific metric overrides a global BGP or group-specific metric.
- A metric defined with a routing policy overrides a metric defined with the `metric-out` statement.
- If any metric is defined, it overrides a metric received in a route.

- If the received route does not have an associated MED metric, and if you do not explicitly configure a metric value, no metric is advertised. When you do not explicitly configure a metric value, the MED value is equivalent to zero (0) when advertising an active route.

Because the AS path rather than the number of hops between hosts is the primary criterion for BGP route selection, an AS with multiple connections to a peer AS can have multiple equivalent AS paths. When the routing table contains two routes to the same host in a neighboring AS, a MED metric assigned to each route can determine which to include in the forwarding table. The MED metric you assign can force traffic through a particular exit point in an AS.

Figure 28 on page 354 illustrates how MED metrics are used to determine route selection.

Figure 28: Default MED Example

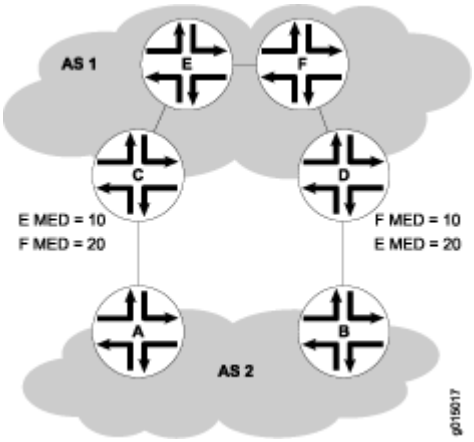


Figure 28 on page 354 shows AS 1 and AS 2 connected by two separate BGP links to Routers C and D. Host E in AS 1 is located nearer to Router C. Host F, also in AS 1, is located nearer to Router D. Because the AS paths are equivalent, two routes exist for each host, one through Router C and one through Router D. To force all traffic destined for Host E through Router C, the network administrator for AS 1 assigns a MED metric for each router to Host E at its exit point. A MED metric of 10 is assigned to the route to Host E through Router C, and a MED metric of 20 is assigned to the route to Host E through Router D. BGP routers in AS 2 select the route with the lower MED metric for the forwarding table.

By default, only the MEDs of routes that have the same peer ASs are compared. However, you can configure the routing table path selection options listed in Table 5 on page 355 to compare MEDs in different ways. The MED options are not mutually exclusive and can be configured in combination or independently. For the MED options to take effect, you must configure them uniformly all through your network. The MED option or options you configure determine the route selected. Thus we recommend that you carefully evaluate your network for preferred routes before configuring the MED options.

Table 5: MED Options for Routing Table Path Selection

Option (Name)	Function	Use
Always comparing MEDs (`always-compare-med`)	Ensures that the MEDs for paths from peers in different ASs are always compared in the route selection process.	Useful when all enterprises participating in a network agree on a uniform policy for setting MEDs. For example, in a network shared by two ISPs, both must agree that a certain path is the better path to configure the MED values correctly.
Adding IGP cost to MED (`med-plus-igp`)	Before comparing MED values for path selection, adds to the MED the cost of the IGP route to the BGP next-hop destination. This option replaces the MED value for the router, but does not affect the IGP metric comparison. As a result, when multiple routes have the same value after the MED-plus-IGP comparison, and route selection continues, the IGP route metric is also compared, even though it was added to the MED value and compared earlier in the selection process.	Useful when the downstream AS requires the complete cost of a certain route that is received across multiple ASs.

Table 5: MED Options for Routing Table Path Selection *(Continued)*

Option (Name)	Function	Use
Applying Cisco IOS nondeterministic behavior (`cisco-non-deterministic`)	Specifies the nondeterministic behavior of the Cisco IOS software: • The active path is always first. All non-active but eligible paths follow the active path and are maintained in the order in which they were received. Ineligible paths remain at the end of the list. • When a new path is added to the routing table, path comparisons are made among all routes, including those paths that must never be selected because they lose the MED tie-breaking rule.	We recommend that you do not configure this option, because the nondeterministic behavior sometimes prevents the system from properly comparing the MEDs between paths.

SEE ALSO

Example: Configuring the MED Using Route Filters | 376

Example: Creating a Named Scope for Multicast Scoping

Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates | 397

Example: Configuring the MED Attribute That Determines the Exit Point in an AS

IN THIS SECTION

Requirements | 357

Overview | 357

Configuration | 358
Verification | 373

This example shows how to configure a multiple exit discriminator (MED) metric to advertise in BGP update messages.

Requirements

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

Overview

To directly configure a MED metric to advertise in BGP update messages, include the `metric-out` statement:

```
metric-out (metric | minimum-igp offset | igp delay-med-update | offset);
```

metric is the primary metric on all routes sent to peers. It can be a value in the range from 0 through 4,294,967,295 (2^{32} – 1).

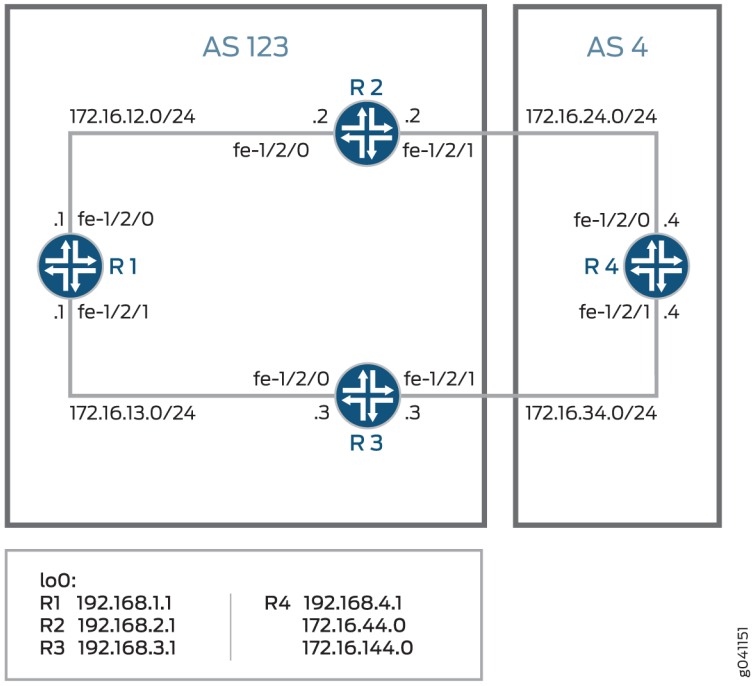
The following optional settings are also supported:

- `minimum-igp`—Sets the metric to the minimum metric value calculated in the interior gateway protocol (IGP) to get to the BGP next hop. If a newly calculated metric is greater than the minimum metric value, the metric value remains unchanged. If a newly calculated metric is lower, the metric value is lowered to that value.
- `igp`—Sets the metric to the most recent metric value calculated in the IGP to get to the BGP next hop.
- `delay-med-update`—Delays sending MED updates when the MED value increases. Include the `delay-med-update` statement when you configure the `igp` statement. The default interval to delay sending updates, unless the MED is lower or another attribute associated with the route has changed is 10 minutes. Include the `med-igp-update-interval` *minutes* statement at the `[edit routing-options]` hierarchy level to modify the default interval.
- *offset*—Specifies a value for *offset* to increase or decrease the metric that is used from the metric value calculated in the IGP. The metric value is offset by the value specified. The metric calculated in the IGP (by specifying either `igp` or `igp-minimum`) is increased if the *offset* value is positive. The metric calculated in the IGP (by specifying either `igp` or `igp-minimum`) is decreased if the *offset* value is negative.

offset can be a value in the range from -2^{31} through $2^{31} - 1$. Note that the adjusted metric can never go below 0 or above $2^{32} - 1$.

Figure 29 on page 358 shows a typical network with internal peer sessions and multiple exit points to a neighboring autonomous system (AS).

Figure 29: Typical Network with IBGP Sessions and Multiple Exit Points



Device R4 has multiple loopback interfaces configured to simulate advertised prefixes. The extra loopback interface addresses are 44.44.44.44/32 and 144.144.144.144/32. This example shows how to configure Device R4 to advertise a MED value of 30 to Device R3 and a MED value of 20 to Device R2. This causes all of the devices in AS 123 to prefer the path through Device R2 to reach AS 4.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **359**
- Configuring Device R1 | **361**
- Configuring Device R2 | **364**
- Configuring Device R3 | **367**

- Configuring Device R4 | 370

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 12.12.12.1/24
set interfaces fe-1/2/1 unit 2 family inet address 13.13.13.1/24
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.1.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.1.1
```

Device R2

```
set interfaces fe-1/2/0 unit 3 family inet address 12.12.12.2/24
set interfaces fe-1/2/1 unit 4 family inet address 24.24.24.2/24
set interfaces lo0 unit 2 family inet address 192.168.2.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.2.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
```

```
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 24.24.24.4
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.2.1
```

Device R3

```
set interfaces fe-1/2/0 unit 5 family inet address 13.13.13.3/24
set interfaces fe-1/2/1 unit 6 family inet address 34.34.34.3/24
set interfaces lo0 unit 3 family inet address 192.168.3.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.3.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 34.34.34.4
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.5
set protocols ospf area 0.0.0.0 interface fe-1/2/1.6
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.3.1
```

Device R4

```
set interfaces fe-1/2/0 unit 7 family inet address 24.24.24.4/24
set interfaces fe-1/2/1 unit 8 family inet address 34.34.34.4/24
set interfaces lo0 unit 4 family inet address 192.168.4.1/32
set interfaces lo0 unit 4 family inet address 44.44.44.44/32
set interfaces lo0 unit 4 family inet address 144.144.144.144/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
```

```
set protocols bgp group external peer-as 123
set protocols bgp group external neighbor 34.34.34.3 metric-out 30
set protocols bgp group external neighbor 24.24.24.2 metric-out 20
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 4
set routing-options router-id 192.168.4.1
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 1]
user@R1# set family inet address 12.12.12.1/24
[edit interfaces fe-1/2/1 unit 2]
user@R1# set family inet address 13.13.13.1/24
[edit interfaces lo0 unit 1]
user@R1# set family inet address 192.168.1.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set local-address 192.168.1.1
user@R1# set export send-direct
user@R1# set neighbor 192.168.2.1
user@R1# set neighbor 192.168.3.1
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface lo0.1 passive
```

```
user@R1# set interface fe-1/2/0.1
user@R1# set interface fe-1/2/1.2
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

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

```
[edit routing-options]
user@R1# set autonomous-system 123
user@R1# set router-id 192.168.1.1
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 12.12.12.1/24;
        }
    }
}
fe-1/2/1 {
    unit 2 {
        family inet {
            address 13.13.13.1/24;
        }
    }
}
lo0 {
```

```
    unit 1 {
        family inet {
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.1.1;
        export send-direct;
        neighbor 192.168.2.1;
        neighbor 192.168.3.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface fe-1/2/0.1;
        interface fe-1/2/1.2;
    }
}
```

```
user@R1# show routing-options
autonomous-system 123;
router-id 192.168.1.1;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 3]
user@R2# set family inet address 12.12.12.21/24
[edit interfaces fe-1/2/1 unit 4]
user@R2# set family inet address 24.24.24.2/24
[edit interfaces lo0 unit 2]
user@R2# set family inet address 192.168.2.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R2# set type internal
user@R2# set local-address 192.168.2.1
user@R2# set export send-direct
user@R2# set neighbor 192.168.1.1
user@R2# set neighbor 192.168.3.1
[edit protocols bgp group external]
user@R2# set type external
user@R2# set export send-direct
user@R2# set peer-as 4
user@R2# set neighbor 24.24.24.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.2 passive
```

```
user@R2# set interface fe-1/2/0.3
user@R2# set interface fe-1/2/1.4
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

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

```
[edit routing-options]
user@R2# set autonomous-system 123
user@R2# set router-id 192.168.2.1
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 3 {
        family inet {
            address 12.12.12.2/24;
        }
    }
}
fe-1/2/1 {
    unit 4 {
        family inet {
            address 24.24.24.2/24;
        }
    }
}
lo0 {
```

```
    unit 2 {
        family inet {
            address 192.168.2.1/32;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R2# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.2.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.3.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 24.24.24.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface fe-1/2/0.3;
        interface fe-1/2/1.4;
```

```
    }
}
```

```
user@R2# show routing-options
autonomous-system 123;
router-id 192.168.2.1;
```

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

Configuring Device R3

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 Junos OS CLI User Guide.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 5]
user@R3# set family inet address 13.13.13.3/24
[edit interfaces fe-1/2/1 unit 6]
user@R3# set family inet address 34.34.34.3/24
[edit interfaces lo0 unit 3]
user@R3# set family inet address 192.168.3.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R3# set type internal
user@R3# set local-address 192.168.3.1
user@R3# set export send-direct
user@R3# set neighbor 192.168.1.1
user@R3# set neighbor 192.168.2.1
[edit protocols bgp group external]
user@R3# set type external
user@R3# set export send-direct
```

```
user@R3# set peer-as 4
user@R3# set neighbor 34.34.34.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R3# set interface lo0.3 passive
user@R3# set interface fe-1/2/0.5
user@R3# set interface fe-1/2/1.6
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R3# set from protocol direct
user@R3# set then accept
```

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

```
[edit routing-options]
user@R3# set autonomous-system 123
user@R3# set router-id 192.168.3.1
```

Results

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

```
user@R3# show interfaces
fe-1/2/0 {
    unit 5 {
        family inet {
            address 13.13.13.3/24;
        }
    }
}
```

```
fe-1/2/1 {
    unit 6 {
        family inet {
            address 34.34.34.3/24;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.3.1/32;
        }
    }
}
```

```
user@R3# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R3# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.3.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.2.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 34.34.34.4;
    }
}
ospf {
```

```
    area 0.0.0.0 {
        interface lo0.3 {
            passive;
        }
        interface fe-1/2/0.5;
        interface fe-1/2/1.6;
    }
}
```

```
user@R3# show routing-options
autonomous-system 123;
router-id 192.168.3.1;
```

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

Configuring Device R4

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 Junos OS CLI User Guide.

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 7]
user@R4# set family inet address 24.24.24.4/24
[edit interfaces fe-1/2/1 unit 8]
user@R4# set family inet address 34.34.34.4/24
[edit interfaces lo0 unit 4]
user@R4# set family inet address 192.168.4.1/32
user@R4# set family inet address 44.44.44.44/32
user@R4# set family inet address 144.144.144.144/32
```

   Device R4 has multiple loopback interface addresses to simulate advertised prefixes.

2. Configure a policy that accepts direct routes.

Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R4# set from protocol direct
user@R4# set then accept
```

3. Configure BGP.

```
[edit protocols bgp group external]
user@R4# set type external
user@R4# set export send-direct
user@R4# set peer-as 123
```

4. Configure a MED value of 30 for neighbor Device R3, and a MED value of 20 for neighbor Device R2.

```
[edit protocols bgp group external]
user@R4# set neighbor 34.34.34.3 metric-out 30
user@R4# set neighbor 24.24.24.2 metric-out 20
```

This configuration causes autonomous system (AS) 123 (of which Device R1, Device R2, and Device R3 are members) to prefer the path through Device R2 to reach AS 4.

5. Configure the router ID and AS number.

```
[edit routing-options]
user@R4# set autonomous-system 4
user@R4# set router-id 192.168.4.1
```

Results

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

```
user@R4# show interfaces
fe-1/2/0 {
```

```
    unit 7 {
        family inet {
            address 24.24.24.4/24;
        }
    }
}
fe-1/2/1 {
    unit 8 {
        family inet {
            address 34.34.34.4/24;
        }
    }
}
lo0 {
    unit 4 {
        family inet {
            address 192.168.4.1/32;
             address 44.44.44.44/32;
             address 144.144.144.144/32;
        }
    }
}
```

```
user@R4# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R4# show protocols
bgp {
    group external {
        type external;
        export send-direct;
        peer-as 123;
        neighbor 34.34.34.3 {
            metric-out 30;
        }
```

```
        neighbor 24.24.24.2 {
            metric-out 20;
        }
    }
}
```

```
user@R4# show routing-options
autonomous-system 4;
router-id 192.168.4.1;
```

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

Verification

IN THIS SECTION

- Checking the Active Path From Device R1 to Device R4 | 373
- Verifying That Device R4 Is Sending Its Routes Correctly | 374

Confirm that the configuration is working properly.

Checking the Active Path From Device R1 to Device R4

Purpose

Verify that the active path goes through Device R2.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 13 destinations, 19 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

12.12.12.0/24       [BGP/170] 3d 22:52:38, localpref 100, from 192.168.2.1
                      AS path: I
```

```
                    > to 12.12.12.2 via fe-1/2/0.1
13.13.13.0/24       [BGP/170] 3d 03:15:16, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
24.24.24.0/24       [BGP/170] 3d 22:52:38, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
34.34.34.0/24       [BGP/170] 3d 03:15:16, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
44.44.44.44/32     *[BGP/170] 01:41:11, MED 20, localpref 100, from 192.168.2.1
                      AS path: 4 I
                    > to 12.12.12.2 via fe-1/2/0.1
144.144.144.144/32 *[BGP/170] 00:08:13, MED 20, localpref 100, from 192.168.2.1
                      AS path: 4 I
                    > to 12.12.12.2 via fe-1/2/0.1
192.168.2.1/32      [BGP/170] 3d 22:52:38, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 12.12.12.2 via fe-1/2/0.1
192.168.3.1/32      [BGP/170] 3d 03:15:16, localpref 100, from 192.168.3.1
                      AS path: I
                    > to 13.13.13.3 via fe-1/2/1.2
192.168.4.1/32     *[BGP/170] 01:41:11, MED 20, localpref 100, from 192.168.2.1
                      AS path: 4 I
                    > to 12.12.12.2 via fe-1/2/0.1
```

Meaning

The asterisk (*) shows that the preferred path is through Device R2. The reason for the path selection is listed as MED 20.

Verifying That Device R4 Is Sending Its Routes Correctly

Purpose

Make sure that Device R4 is sending update messages with a value of 20 to Device R2 and a value of 30 to Device R3.

Action

From operational mode, enter the `show route advertising-protocol bgp 24.24.24.2` command.

```
user@R4> show route advertising-protocol bgp 24.24.24.2
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 24.24.24.0/24           Self                 20                 I
* 34.34.34.0/24           Self                 20                 I
* 44.44.44.44/32          Self                 20                 I
* 144.144.144.144/32      Self                 20                 I
* 192.168.4.1/32          Self                 20                 I
```

```
user@R4> show route advertising-protocol bgp 34.34.34.3
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 24.24.24.0/24           Self                 30                 I
* 34.34.34.0/24           Self                 30                 I
* 44.44.44.44/32          Self                 30                 I
* 144.144.144.144/32      Self                 30                 I
* 192.168.4.1/32          Self                 30                 I
```

Meaning

The MED column shows that Device R4 is sending the correct MED values to its two external BGP (EBGP) neighbors.

SEE ALSO

- Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates | 397
- Understanding BGP Path Selection | 12
- Understanding External BGP Peering Sessions | 26
- BGP Configuration Overview | 24

Example: Configuring the MED Using Route Filters

IN THIS SECTION

- Requirements | 376
- Overview | 376
- Configuration | 377
- Verification | 393

This example shows how to configure a policy that uses route filters to modify the multiple exit discriminator (MED) metric to advertise in BGP update messages.

Requirements

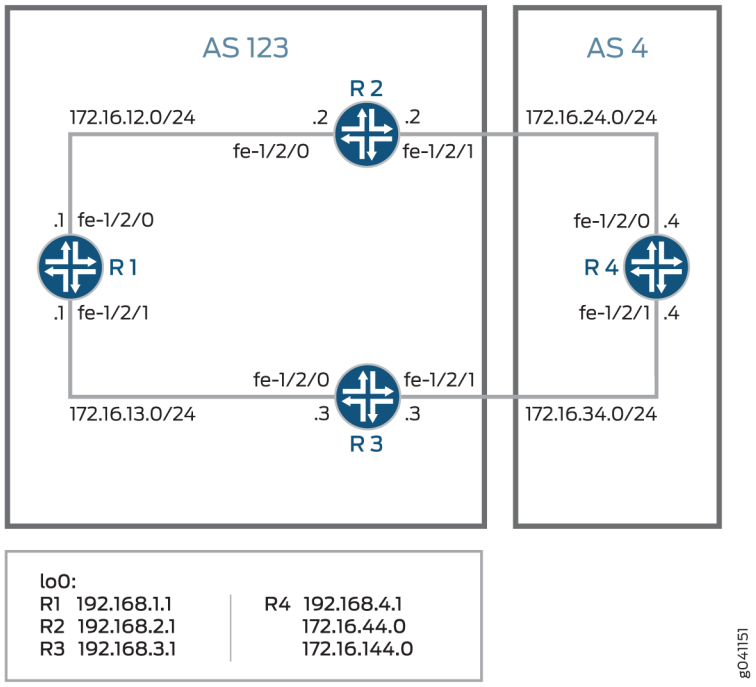
No special configuration beyond device initialization is required before you configure this example.

Overview

To configure a route-filter policy that modifies the advertised MED metric in BGP update messages, include the `metric` statement in the policy action.

Figure 30 on page 377 shows a typical network with internal peer sessions and multiple exit points to a neighboring autonomous system (AS).

Figure 30: Typical Network with IBGP Sessions and Multiple Exit Points



Device R4 has multiple loopback interfaces configured to simulate advertised prefixes. The extra loopback interface addresses are 172.16.44.0/32 and 172.16.144.0/32. This example shows how to configure Device R4 to advertise a MED value of 30 to Device R3 for all routes except 172.16.144.0. For 172.16.144.0, a MED value of 10 is advertised to Device 3. A MED value of 20 is advertised to Device R2, regardless of the route prefix.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **378**
- Configuring Device R1 | **380**
- Configuring Device R2 | **383**
- Configuring Device R3 | **386**
- Configuring Device R4 | **389**

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 172.16.12.1/24
set interfaces fe-1/2/1 unit 2 family inet address 172.16.13.1/24
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.1.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.1.1
```

Device R2

```
set interfaces fe-1/2/0 unit 3 family inet address 172.16.12.2/24
set interfaces fe-1/2/1 unit 4 family inet address 172.16.24.2/24
set interfaces lo0 unit 2 family inet address 192.168.2.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.2.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 172.16.24.4
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set policy-options policy-statement send-direct term 1 from protocol direct
```

```
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.2.1
```

Device R3

```
set interfaces fe-1/2/0 unit 5 family inet address 172.16.13.3/24
set interfaces fe-1/2/1 unit 6 family inet address 172.16.34.3/24
set interfaces lo0 unit 3 family inet address 192.168.3.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.3.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 172.16.34.4
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.5
set protocols ospf area 0.0.0.0 interface fe-1/2/1.6
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.3.1
```

Device R4

```
set interfaces fe-1/2/0 unit 7 family inet address 172.16.24.4/24
set interfaces fe-1/2/1 unit 8 family inet address 172.16.34.4/24
set interfaces lo0 unit 4 family inet address 192.168.4.1/32
set interfaces lo0 unit 4 family inet address 172.16.44.0/32
set interfaces lo0 unit 4 family inet address 172.16.144.0/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 123
set protocols bgp group external neighbor 172.16.34.3 export med-10
set protocols bgp group external neighbor 172.16.34.3 export med-30
set protocols bgp group external neighbor 172.16.24.2 metric-out 20
set policy-options policy-statement med-10 from route-filter 172.16.144.0/32 exact
set policy-options policy-statement med-10 then metric 10
```

```
set policy-options policy-statement med-10 then accept
set policy-options policy-statement med-30 from route-filter 0.0.0.0/0 longer
set policy-options policy-statement med-30 then metric 30
set policy-options policy-statement med-30 then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 4
set routing-options router-id 192.168.4.1
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 1]
user@R1# set family inet address 172.16.12.1/24
[edit interfaces fe-1/2/1 unit 2]
user@R1# set family inet address 172.16.13.1/24
[edit interfaces lo0 unit 1]
user@R1# set family inet address 192.168.1.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set local-address 192.168.1.1
user@R1# set export send-direct
user@R1# set neighbor 192.168.2.1
user@R1# set neighbor 192.168.3.1
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface lo0.1 passive
user@R1# set interface fe-1/2/0.1
user@R1# set interface fe-1/2/1.2
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

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

```
[edit routing-options]
user@R1# set autonomous-system 123
user@R1# set router-id 192.168.1.1
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 172.16.12.1/24;
        }
    }
}
fe-1/2/1 {
    unit 2 {
        family inet {
```

```
            address 172.16.13.1/24;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.1.1;
        export send-direct;
        neighbor 192.168.2.1;
        neighbor 192.168.3.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface fe-1/2/0.1;
        interface fe-1/2/1.2;
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
```

```
    }
}
```

```
user@R1# show routing-options
autonomous-system 123;
router-id 192.168.1.1;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 3]
user@R2# set family inet address 172.16.12.21/24
[edit interfaces fe-1/2/1 unit 4]
user@R2# set family inet address 172.16.24.2/24
[edit interfaces lo0 unit 2]
user@R2# set family inet address 192.168.2.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R2# set type internal
user@R2# set local-address 192.168.2.1
user@R2# set export send-direct
user@R2# set neighbor 192.168.1.1
user@R2# set neighbor 192.168.3.1
[edit protocols bgp group external]
user@R2# set type external
user@R2# set export send-direct
```

```
user@R2# set peer-as 4
user@R2# set neighbor 172.16.24.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.2 passive
user@R2# set interface fe-1/2/0.3
user@R2# set interface fe-1/2/1.4
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

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

```
[edit routing-options]
user@R2# set autonomous-system 123
user@R2# set router-id 192.168.2.1
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
fe-1/2/0 {
    unit 3 {
        family inet {
            address 172.16.12.2/24;
        }
    }
}
```

```
fe-1/2/1 {
    unit 4 {
        family inet {
            address 172.16.24.2/24;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.168.2.1/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.2.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.3.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 172.16.24.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface fe-1/2/0.3;
        interface fe-1/2/1.4;
```

```
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R2# show routing-options
autonomous-system 123;
router-id 192.168.2.1;
```

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

Configuring Device R3

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 Junos OS CLI User Guide.

To configure Device R3:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 5]
user@R3# set family inet address 172.16.13.3/24
[edit interfaces fe-1/2/1 unit 6]
user@R3# set family inet address 172.16.34.3/24
[edit interfaces lo0 unit 3]
user@R3# set family inet address 192.168.3.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R3# set type internal
user@R3# set local-address 192.168.3.1
user@R3# set export send-direct
user@R3# set neighbor 192.168.1.1
user@R3# set neighbor 192.168.2.1
[edit protocols bgp group external]
user@R3# set type external
user@R3# set export send-direct
user@R3# set peer-as 4
user@R3# set neighbor 172.16.34.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R3# set interface lo0.3 passive
user@R3# set interface fe-1/2/0.5
user@R3# set interface fe-1/2/1.6
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R3# set from protocol direct
user@R3# set then accept
```

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

```
[edit routing-options]
user@R3# set autonomous-system 123
user@R3# set router-id 192.168.3.1
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R3# show interfaces
fe-1/2/0 {
    unit 5 {
        family inet {
            address 172.16.13.3/24;
        }
    }
}
fe-1/2/1 {
    unit 6 {
        family inet {
            address 172.16.34.3/24;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.3.1/32;
        }
    }
}
```

```
user@R3# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.3.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.2.1;
    }
    group external {
        type external;
        export send-direct;
```

```
        peer-as 4;
        neighbor 172.16.34.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.3 {
            passive;
        }
        interface fe-1/2/0.5;
        interface fe-1/2/1.6;
    }
}
```

```
user@R3# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R3# show routing-options
autonomous-system 123;
router-id 192.168.3.1;
```

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

Configuring Device R4

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 Junos OS CLI User Guide.

To configure Device R4:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 7]
user@R4# set family inet address 172.16.24.4/24
[edit interfaces fe-1/2/1 unit 8]
user@R4# set family inet address 172.16.34.4/24
[edit interfaces lo0 unit 4]
user@R4# set family inet address 192.168.4.1/32
user@R4# set family inet address 172.16.44.0/32
user@R4# set family inet address 172.16.144.0/32
```

   Device R4 has multiple loopback interface addresses to simulate advertised prefixes.

2. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R4# set from protocol direct
user@R4# set then accept
```

3. Configure BGP.

```
[edit protocols bgp group external]
user@R4# set type external
user@R4# set export send-direct
user@R4# set peer-as 123
```

4. Configure the two MED policies.

```
[edit policy-options]
set policy-statement med-10 from route-filter 172.16.144.0/32 exact
set policy-statement med-10 then metric 10
set policy-statement med-10 then accept
set policy-statement med-30 from route-filter 0.0.0.0/0 longer
set policy-statement med-30 then metric 30
set policy-statement med-30 then accept
```

5. Configure the two EBGP neighbors, applying the two MED policies to Device R3, and a MED value of 20 to Device R2.

```
[edit protocols bgp group external]
user@R4# set neighbor 172.16.34.3 export med-10
user@R4# set neighbor 172.16.34.3 export med-30
user@R4# set neighbor 172.16.24.2 metric-out 20
```

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

```
[edit routing-options]
user@R4# set autonomous-system 4
user@R4# set router-id 192.168.4.1
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R4# show interfaces
fe-1/2/0 {
    unit 7 {
        family inet {
            address 172.16.24.4/24;
        }
    }
}
fe-1/2/1 {
    unit 8 {
        family inet {
            address 172.16.34.4/24;
        }
    }
}
lo0 {
    unit 4 {
        family inet {
            address 192.168.4.1/32;
```

```
            address 172.16.44.0/32;
            address 172.16.144.0/32;
        }
    }
}
```

```
user@R4# show protocols
bgp {
    group external {
        type external;
        export send-direct;
        peer-as 123;
        neighbor 172.16.24.2 {
            metric-out 20;
        }
        neighbor 172.16.34.3 {
            export [ med-10 med-30 ];
        }
    }
}
```

```
user@R4# show policy-options
policy-statement med-10 {
    from {
        route-filter 172.16.144.0/32 exact;
    }
    then {
        metric 10;
        accept;
    }
}
policy-statement med-30 {
    from {
        route-filter 0.0.0.0/0 longer;
    }
    then {
        metric 30;
        accept;
    }
}
```

```
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R4# show routing-options
autonomous-system 4;
router-id 192.168.4.1;
```

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

Verification

IN THIS SECTION

- Checking the Active Path from Device R1 to Device R4 | **393**
- Verifying That Device R4 Is Sending Its Routes Correctly | **394**

Confirm that the configuration is working properly.

Checking the Active Path from Device R1 to Device R4

Purpose

Verify that the active path goes through Device R2.

Action

From operational mode, enter the `show route protocol bgp` command.

```
user@R1> show route protocol bgp
inet.0: 13 destinations, 19 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.12.0/24       [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
```

```
                    AS path: I
                  > to 172.16.12.2 via fe-1/2/0.1
172.16.13.0/24      [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
                    AS path: I
                  > to 172.16.13.3 via fe-1/2/1.2
172.16.24.0/24      [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
                    AS path: I
                  > to 172.16.12.2 via fe-1/2/0.1
172.16.34.0/24      [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
                    AS path: I
                  > to 172.16.13.3 via fe-1/2/1.2
172.16.44.0/32    *[BGP/170] 00:06:03, MED 20, localpref 100, from 192.168.2.1
                    AS path: 4 I
                  > to 172.16.12.2 via fe-1/2/0.1
172.16.144.0/32 *[BGP/170] 00:06:03, MED 10, localpref 100, from 192.168.3.1
                    AS path: 4 I
                  > to 172.16.13.3 via fe-1/2/1.2
192.168.2.1/32     [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
                    AS path: I
                  > to 172.16.12.2 via fe-1/2/0.1
192.168.3.1/32     [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
                    AS path: I
                  > to 172.16.13.3 via fe-1/2/1.2
192.168.4.1/32    *[BGP/170] 00:06:03, MED 20, localpref 100, from 192.168.2.1
                    AS path: 4 I
                  > to 172.16.12.2 via fe-1/2/0.1
```

Meaning

The output shows that the preferred path to the routes advertised by Device R4 is through Device R2 for all routes except 172.16.144.0/32. For 172.16.144.0/32, the preferred path is through Device R3.

Verifying That Device R4 Is Sending Its Routes Correctly

Purpose

Make sure that Device R4 is sending update messages with a value of 20 to Device R2 and a value of 30 to Device R3.

Action

From operational mode, enter the `show route advertising-protocol bgp` command.

```
user@R4> show route advertising-protocol bgp 172.16.24.2
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.24.0/24           Self                 20                 I
* 172.16.34.0/24           Self                 20                 I
* 172.16.44.0/32           Self                 20                 I
* 172.16.144.0/32      Self                 20                 I
* 192.168.4.1/32           Self                 20                 I
```

```
user@R4> show route advertising-protocol bgp 172.16.34.3
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.24.0/24           Self                 30                 I
* 172.16.34.0/24           Self                 30                 I
* 172.16.44.0/32           Self                 30                 I
* 172.16.144.0/32      Self                 10                 I
* 192.168.4.1/32           Self                 30                 I
```

Meaning

The MED column shows that Device R4 is sending the correct MED values to its two EBGP neighbors.

SEE ALSO

- Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates
- *Understanding Route Filters for Use in Routing Policy Match Conditions*
- Understanding BGP Path Selection | **12**
- Understanding External BGP Peering Sessions

Example: Configuring the MED Using Communities

Set the multiple exit discriminator (MED) metric to 20 for all routes from a particular community.

```
[edit]
routing-options {
    router-id 10.0.0.1;
    autonomous-system 23;
}
policy-options {
    policy-statement from-otago {
        from community otago;
        then metric 20;
    }
    community otago members [56:2379 23:46944];
}
protocols {
    bgp {
        import from-otago;
        group 23 {
            type external;
            peer-as 56;
            neighbor 192.168.0.1 {
                traceoptions {
                    file bgp-log-peer;
                    flag packets;
                }
                log-updown;
            }
        }
    }
}
```

Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates

IN THIS SECTION

- Requirements | 397
- Overview | 397
- Configuration | 400
- Verification | 409

This example shows how to associate the multiple exit discriminator (MED) path attribute with the interior gateway protocol (IGP) metric, and configure a timer to delay update of the MED attribute.

Requirements

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

Overview

BGP can be configured to advertise the MED attribute for a route based on the IGP distance of its internal BGP (IBGP) route next-hop. The IGP metric enables internal routing to follow the shortest path according to the administrative setup. In some deployments, it might be ideal to communicate IGP shortest-path knowledge to external BGP (EBGP) peers in a neighboring autonomous system (AS). This allows those EBGP peers to forward traffic into your AS using the shortest paths possible.

Routes learned from an EBGP peer usually have a next hop on a directly connected interface, and thus the IGP value is equal to zero. Zero is the value advertised. The IGP metric is a nonzero value when a BGP peer sends third-party next hops that require the local system to perform next-hop resolution—IBGP configurations, configurations within confederation peers, or EBGP configurations that include the `multihop` statement. In these scenarios, it might make sense to associate the MED value with the IGP metric by including the `metric-out minimum-igp` or `metric-out igp` option.

The drawback of associating the MED with the IGP metric is the risk of excessive route advertisements when there are IGP instabilities in the network. Configuring a delay for the MED update provides a mechanism to reduce route advertisements in such scenarios. The delay works by slowing down MED updates when the IGP metric for the next hop changes. The approach uses a timer to periodically advertise MED updates. When the timer expires, the MED attribute for routes with `metric-out igp delay-updates` configured is updated to the current IGP metric of the next hop. The BGP-enabled device sends out advertisements for routes for which the MED attribute has changed.

The `delay-updates` option identifies the BGP groups (or peers) for which the MED updates must be suppressed. The time for advertising MED updates is set to 10 minutes by default. You can increase the interval up to 600 minutes by including the `med-igp-update-interval` statement in the `routing-options` configuration.

NOTE: If you have nonstop active routing (NSR) enabled and a switchover occurs, the delayed MED updates might be advertised as soon as the switchover occurs.

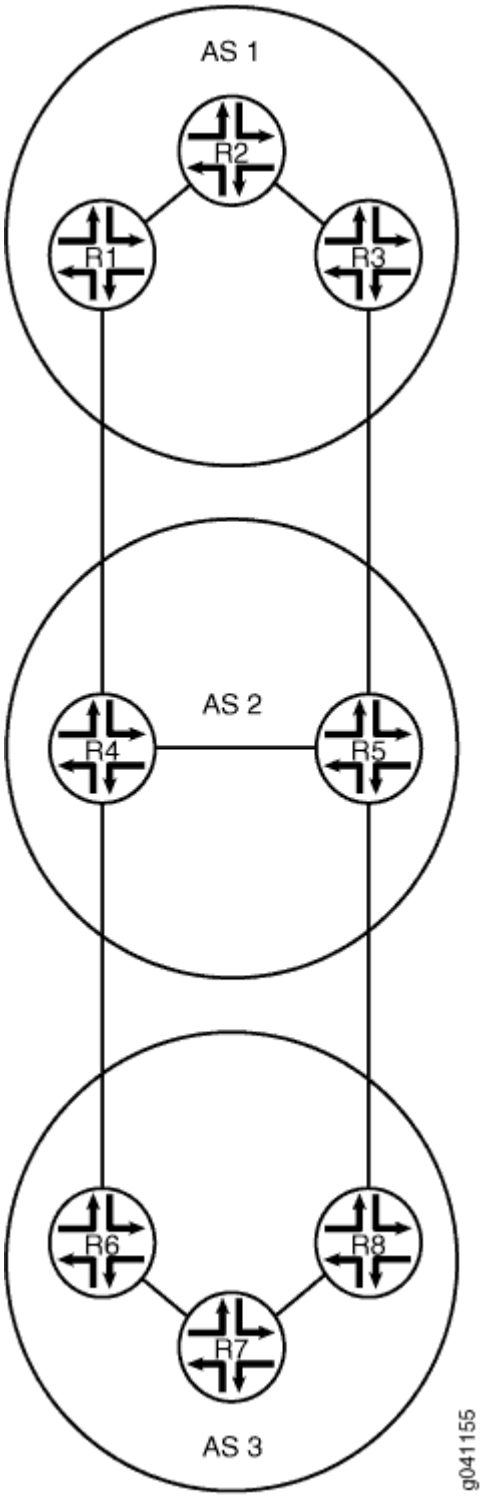
When you configure the `metric-out igp` option, the IGP metric directly tracks the IGP cost to the IBGP peer. When the IGP cost goes down, so does the advertised MED value. Conversely, when the IGP cost goes up, the MED value goes up as well.

When you configure the `metric-out minimum-igp` option, the advertised MED value changes only when the IGP cost to the IBGP peer goes down. An increase in the IGP cost does not affect the MED value. The router monitors and remembers the lowest IGP cost until the routing process (rpd) is restarted. The BGP peer sends an update only if the MED is lower than the previously advertised value or another attribute associated with the route has changed, or if the BGP peer is responding to a refresh route request.

This example uses the `metric` statement in the OSPF configuration to demonstrate that when the IGP metric changes, the MED also changes after the configured delay interval. The OSPF metric can range from 1 through 65,535.

Figure 31 on page 399 shows the sample topology.

Figure 31: Topology for Delaying the MED Update



In this example, the MED value advertised by Device R1 is associated with the IGP running in AS 1. The MED value advertised by Device R1 impacts the decisions of the neighboring AS (AS 2) when AS 2 is forwarding traffic into AS 1.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **400**
- Configuring Device R1 | **405**

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 R1

```
set interfaces fe-1/2/0 unit 2 description R1->R2
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 7 description R1->R4
set interfaces fe-1/2/1 unit 7 family inet address 172.16.0.1/30
set interfaces lo0 unit 1 family inet address 192.168.0.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.2
set protocols bgp group internal neighbor 192.168.0.3
set protocols bgp group external type external
set protocols bgp group external metric-out igp delay-med-update
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 2
set protocols bgp group external neighbor 172.16.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.2 metric 600
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options med-igp-update-interval 12
```

```
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device R2

```
set interfaces fe-1/2/0 unit 1 description R2->R1
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 4 description R2->R3
set interfaces fe-1/2/1 unit 4 family inet address 10.0.2.2/30
set interfaces lo0 unit 2 family inet address 192.168.0.2/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.2
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.1
set protocols bgp group internal neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 1
```

Device R3

```
set interfaces fe-1/2/0 unit 3 description R3->R2
set interfaces fe-1/2/0 unit 3 family inet address 10.0.2.1/30
set interfaces fe-1/2/1 unit 5 description R3->R5
set interfaces fe-1/2/1 unit 5 family inet address 172.16.0.5/30
set interfaces lo0 unit 3 family inet address 192.168.0.3/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.3
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.1
set protocols bgp group internal neighbor 192.168.0.2
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 2
set protocols bgp group external neighbor 172.16.0.6
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set protocols ospf area 0.0.0.0 interface lo0.3 passive
```

```
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 1
```

Device R4

```
set interfaces fe-1/2/0 unit 8 description R4->R1
set interfaces fe-1/2/0 unit 8 family inet address 172.16.0.2/30
set interfaces fe-1/2/1 unit 9 description R4->R5
set interfaces fe-1/2/1 unit 9 family inet address 10.0.4.1/30
set interfaces fe-1/2/2 unit 13 description R4->R6
set interfaces fe-1/2/2 unit 13 family inet address 172.16.0.9/30
set interfaces lo0 unit 4 family inet address 192.168.0.4/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.4
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.5
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external neighbor 172.16.0.10 peer-as 3
set protocols bgp group external neighbor 172.16.0.1 peer-as 1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.9
set protocols ospf area 0.0.0.0 interface lo0.4 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.4
set routing-options autonomous-system 2
```

Device R5

```
set interfaces fe-1/2/0 unit 6 description R5->R3
set interfaces fe-1/2/0 unit 6 family inet address 172.16.0.6/30
set interfaces fe-1/2/1 unit 10 description R5->R4
set interfaces fe-1/2/1 unit 10 family inet address 10.0.4.2/30
set interfaces fe-1/2/2 unit 11 description R5->R8
set interfaces fe-1/2/2 unit 11 family inet address 172.16.0.13/30
set interfaces lo0 unit 5 family inet address 192.168.0.5/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.5
set protocols bgp group internal export send-direct
```

```
set protocols bgp group internal neighbor 192.168.0.4
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external neighbor 172.16.0.5 peer-as 1
set protocols bgp group external neighbor 172.16.0.14 peer-as 3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.10
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.5
set routing-options autonomous-system 2
```

Device R6

```
set interfaces fe-1/2/0 unit 14 description R6->R4
set interfaces fe-1/2/0 unit 14 family inet address 172.16.0.10/30
set interfaces fe-1/2/1 unit 15 description R6->R7
set interfaces fe-1/2/1 unit 15 family inet address 10.0.6.1/30
set interfaces lo0 unit 6 family inet address 192.168.0.6/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.6
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.7
set protocols bgp group internal neighbor 192.168.0.8
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 2
set protocols bgp group external neighbor 172.16.0.9 peer-as 2
set protocols ospf area 0.0.0.0 interface fe-1/2/1.15
set protocols ospf area 0.0.0.0 interface lo0.6 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.6
set routing-options autonomous-system 3
```

Device R7

```
set interfaces fe-1/2/0 unit 16 description R7->R6
set interfaces fe-1/2/0 unit 16 family inet address 10.0.6.2/30
set interfaces fe-1/2/1 unit 17 description R7->R8
set interfaces fe-1/2/1 unit 17 family inet address 10.0.7.2/30
```

```
set interfaces lo0 unit 7 family inet address 192.168.0.7/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.7
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.6
set protocols bgp group internal neighbor 192.168.0.8
set protocols ospf area 0.0.0.0 interface fe-1/2/0.16
set protocols ospf area 0.0.0.0 interface fe-1/2/1.17
set protocols ospf area 0.0.0.0 interface lo0.7 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.7
set routing-options autonomous-system 3
```

Device R8

```
set interfaces fe-1/2/0 unit 12 description R8->R5
set interfaces fe-1/2/0 unit 12 family inet address 172.16.0.14/30
set interfaces fe-1/2/1 unit 18 description R8->R7
set interfaces fe-1/2/1 unit 18 family inet address 10.0.7.1/30
set interfaces lo0 unit 8 family inet address 192.168.0.8/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.0.8
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.0.6
set protocols bgp group internal neighbor 192.168.0.7
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 2
set protocols bgp group external neighbor 172.16.0.13 peer-as 2
set protocols ospf area 0.0.0.0 interface fe-1/2/1.18
set protocols ospf area 0.0.0.0 interface lo0.8 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.8
set routing-options autonomous-system 3
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 2]
user@R1# set description R1->R2
user@R1# set family inet address 10.0.0.1/30
[edit interfaces fe-1/2/1 unit 7]
user@R1# set description R1->R4
user@R1# set family inet address 172.16.0.1/30
[edit interfaces lo0 unit 1]
user@R1# set family inet address 192.168.0.1/32
```

2. Configure IBGP.

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set export send-direct
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Configure EBGP.

```
[edit protocols bgp group external]
user@R1# set type external
user@R1# set export send-direct
user@R1# set peer-as 2
user@R1# set neighbor 172.16.0.2
```

4. Associate the MED value with the IGP metric.

```
[edit protocols bgp group external]
user@R1# set metric-out igp delay-med-update
```

   The default for the MED update is 10 minutes when you include the `delay-med-update` option. When you exclude the `delay-med-update` option, the MED update occurs immediately after the IGP metric changes.

5. (Optional) Configure the update interval for the MED update.

```
[edit routing-options]
user@R1# set med-igp-update-interval 12
```

   You can configure the interval from 10 minutes through 600 minutes.

6. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.2 metric 600
user@R1# set interface lo0.1 passive
```

   The `metric` statement is used here to demonstrate what happens when the IGP metric changes.

7. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set then accept
```

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

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 1
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 2 {
        description R1->R2;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/1 {
    unit 7 {
        description R1->R4;
        family inet {
            address 172.16.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
```

```
    }
}
```

```
user@R1# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.0.1;
        export send-direct;
        neighbor 192.168.0.2;
        neighbor 192.168.0.3;
    }
    group external {
        type external;
        metric-out igp delay-med-update;
        export send-direct;
        peer-as 2;
        neighbor 172.16.0.2;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.2 {
            metric 600;
        }
        interface lo0.1 {
            passive;
        }
    }
}
```

```
user@R1# show routing-options
med-igp-update-interval 12;
router-id 192.168.0.1;
autonomous-system 1;
```

If you are done configuring the device, enter `commit` from configuration mode. Repeat the configuration steps on the other devices in the topology, as needed for your network.

Verification

IN THIS SECTION

- Checking the BGP Advertisements | 409
- Verifying That the MED Value Changes When the OSPF Metric Changes | 410
- Testing the minimum-igp Setting | 410

Confirm that the configuration is working properly.

Checking the BGP Advertisements

Purpose

Verify that Device R1 is advertising to Device R4 a BGP MED value that reflects the IGP metric.

Action

From operational mode, enter the `show route advertising-protocol bgp` command.

```
user@R1> show route advertising-protocol bgp 172.16.0.2
inet.0: 19 destinations, 33 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                 0                  I
* 172.16.0.0/30           Self                 0                  I
* 172.16.0.4/30           Self                 601                I
* 192.168.0.1/32          Self                 0                  I
```

Meaning

The 601 value in the MED column shows that the MED value has been updated to reflect the configured OSPF metric.

Verifying That the MED Value Changes When the OSPF Metric Changes

Purpose

Make sure that when you raise the OSPF metric to 700, the MED value is updated to reflect this change.

Action

From configuration mode, enter the `set protocols ospf area 0 interface fe-1/2/0.2 metric 700` command.

```
user@R1# set protocols ospf area 0 interface fe-1/2/0.2 metric 700
user@R1# commit
```

After waiting 12 minutes (the configured delay period), enter the `show route advertising-protocol bgp` command from operational mode.

```
user@R1> show route advertising-protocol bgp 172.16.0.2
inet.0: 19 destinations, 33 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                 0                  I
* 172.16.0.0/30           Self                 0                  I
* 172.16.0.4/30           Self                 701                I
* 192.168.0.1/32          Self                 0                  I
```

Meaning

The 701 value in the MED column shows that the MED value has been updated to reflect the configured OSPF metric.

Testing the minimum-igp Setting

Purpose

Change the configuration to use the `minimum-igp` statement instead of the `igp` statement. When you increase the OSPF metric, the MED value remains unchanged, but when you decrease the OSPF metric, the MED value reflects the new OSPF metric.

Action

From configuration mode, delete the `igp` statement, add the `minimum-igp` statement, and increase the OSPF metric.

```
user@R1# delete protocols bgp group external metric-out igp
user@R1# set protocols bgp group external metric-out minimum-igp
user@R1# set protocols ospf area 0 interface fe-1/2/0.2 metric 800
user@R1# commit
```

From operational mode, enter the `show route advertising-protocol bgp` command to make sure that the MED value does not change.

```
user@R1> show route advertising-protocol bgp 172.16.0.2
inet.0: 19 destinations, 33 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                 0                  I
* 172.16.0.0/30           Self                 0                  I
* 172.16.0.4/30           Self                 701                I
* 192.168.0.1/32          Self                 0                  I
```

From configuration mode, decrease the OSPF metric.

```
user@R1# set protocols ospf area 0 interface fe-1/2/0.2 metric 20
user@R1# commit
```

From operational mode, enter the `show route advertising-protocol bgp` command to make sure that the MED value does change.

```
user@R1> show route advertising-protocol bgp 172.16.0.2
inet.0: 19 destinations, 33 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                 0                  I
* 172.16.0.0/30           Self                 0                  I
* 172.16.0.4/30           Self                 21                 I
* 192.168.0.1/32          Self                 0                  I
```

Meaning

When the `minimum-igp` statement is configured, the MED value changes only when a shorter path is available.

SEE ALSO

Understanding BGP Path Selection | 12

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

BGP Multihop Sessions

IN THIS SECTION

- Understanding EBGP Multihop | 412
- Example: Configuring EBGP Multihop Sessions | 414

Understanding EBGP Multihop

BGP is an exterior gateway protocol (EGP) that is used to exchange routing information among routers in different autonomous systems (ASs). The following are two ways of establishing EBGP multihop between routers:

- When external BGP (EBGP) peers are not directly connected to each other, they must cross one or more non-BGP routers to reach each other.

  Configuring multihop EBGP enables the peers to pass through the other routers to form peer relationships and exchange update messages. This type of configuration is typically used when a Juniper Networks routing device needs to run EBGP with a third-party router that does not allow direct connection of the two EBGP peers. EBGP multihop enables a neighbor connection between two EBGP peers that do not have a direct connection.

- The default behavior for an EBGP connection is to peer over a single physical hop using the physical interface address of the peer. In some cases, it is advantageous to alter this default, one-hop, physical

peering EBGP behavior. One such case is when multiple physical links connect two routers that are to be EBGP peers. In this case, if one of the point-to-point links fails, reachability on the alternate link still exists.

Figure 32: EBGP Multihop Peering

In figure 1, router R1 belongs to AS 1 and router R2 belongs to AS 2. The two physical links between the routers is used for load balancing. The EBGP multihop peering works with one physical link as well.

The following configuration example helps to establish a single BGP peering session across these multiple physical links:

1. Each router must establish the peering session with the loopback address of the remote router. You can configure this session using the `local-address` statement, which alters the peer address header information in the BGP packets.
2. Use the `multihop` statement to alter the default use of the neighbor's physical address. In addition, you can also specify a time-to-live (TTL) value in the BGP packets to control how far they propagate. We use a TTL value of 1 to ensure that the session cannot be established across any other backdoor links in the network.

   NOTE: When multihop is configured, the Junos OS sets the TTL value of 64, by default. A TTL value of 1 is sufficient to enable an EBGP session to the loopback address of a directly connected neighbor.

3. Each router must have IP routing capability to the remote router's loopback address. This capability is often accomplished by using a static route to map the loopback address to the interface physical addresses.

```
[edit protocols bgp group ext-peers]
type external;
local-address 192.168.3.4;
neighbor 172.16.128.1 {
```

```
    multihop ttl 1;
    }
```

```
[edit routing-options]
static {
    route 172.16.128.1 next-hop (10.10.1.1 | 10.10.2.1);
    }
```

SEE ALSO

Example: Configuring EBGP Multihop Sessions on Logical Systems

Example: Configuring EBGP Multihop Sessions

IN THIS SECTION

- Requirements | **414**
- Overview | **414**
- Configuration | **415**
- Verification | **425**

This example shows how to configure an external BGP (EBGP) peer that is more than one hop away from the local router. This type of session is called a *multihop* BGP session.

Requirements

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

Overview

The configuration to enable multihop EBGP sessions requires connectivity between the two EBGP peers. This example uses static routes to provide connectivity between the devices.

Unlike directly connected EBGP sessions in which physical addresses are typically used in the `neighbor` statements, you must use loopback interface addresses for multihop EBGP by specifying the loopback

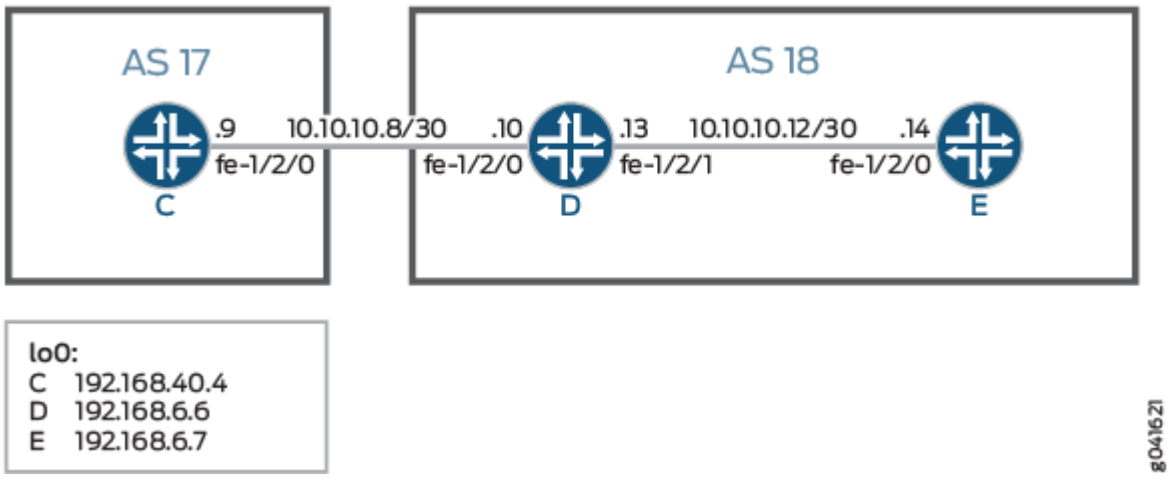
interface address of the indirectly connected peer. In this way, EBGP multihop is similar to internal BGP (IBGP).

Finally, you must add the `multihop` statement. Optionally, you can set a maximum time-to-live (TTL) value with the `ttl` statement. The TTL is carried in the IP header of BGP packets. If you do not specify a TTL value, the system's default maximum TTL value is used. The default TTL value is 64 for multihop EBGP sessions. Another option is to retain the BGP next-hop value for route advertisements by including the `no-nexthop-change` statement.

Figure 33 on page 415 shows a typical EBGP multihop network.

Device C and Device E have an established EBGP session. Device D is not a BGP-enabled device. All of the devices have connectivity via static routes.

Figure 33: Typical Network with EBGP Multihop Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | **416**
- Device C | **417**
- Configuring Device D | **420**
- Configuring Device E | **422**

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 C

```
set interfaces fe-1/2/0 unit 9 description to-D
set interfaces fe-1/2/0 unit 9 family inet address 10.10.10.9/30
set interfaces lo0 unit 3 family inet address 192.168.40.4/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers multihop ttl 2
set protocols bgp group external-peers local-address 192.168.40.4
set protocols bgp group external-peers export send-static
set protocols bgp group external-peers peer-as 18
set protocols bgp group external-peers neighbor 192.168.6.7
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.10.10.14/32 next-hop 10.10.10.10
set routing-options static route 192.168.6.7/32 next-hop 10.10.10.10
set routing-options router-id 192.168.40.4
set routing-options autonomous-system 17
```

Device D

```
set interfaces fe-1/2/0 unit 10 description to-C
set interfaces fe-1/2/0 unit 10 family inet address 10.10.10.10/30
set interfaces fe-1/2/1 unit 13 description to-E
set interfaces fe-1/2/1 unit 13 family inet address 10.10.10.13/30
set interfaces lo0 unit 4 family inet address 192.168.6.6/32
set routing-options static route 192.168.40.4/32 next-hop 10.10.10.9
set routing-options static route 192.168.6.7/32 next-hop 10.10.10.14
set routing-options router-id 192.168.6.6
```

Device E

```
set interfaces fe-1/2/0 unit 14 description to-D
set interfaces fe-1/2/0 unit 14 family inet address 10.10.10.14/30
set interfaces lo0 unit 5 family inet address 192.168.6.7/32
set protocols bgp group external-peers multihop ttl 2
```

```
set protocols bgp group external-peers local-address 192.168.6.7
set protocols bgp group external-peers export send-static
set protocols bgp group external-peers peer-as 17
set protocols bgp group external-peers neighbor 192.168.40.4
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.10.10.8/30 next-hop 10.10.10.13
set routing-options static route 192.168.40.4/32 next-hop 10.10.10.13
set routing-options router-id 192.168.6.7
set routing-options autonomous-system 18
```

Device C

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 Junos OS CLI User Guide.

To configure Device C:

1. Configure the interface to the directly connected device (to-D), and configure the loopback interface.

```
[edit interfaces fe-1/2/0 unit 9]
user@C# set description to-D
user@C# set family inet address 10.10.10.9/30
[edit interfaces lo0 unit 3]
user@C# set family inet address 192.168.40.4/32
```

2. Configure an EBGP session with Device E.

   The `neighbor` statement points to the loopback interface on Device E.

```
[edit protocols bgp group external-peers]
user@C# set type external
user@C# set local-address 192.168.40.4
user@C# set export send-static
user@C# set peer-as 18
user@C# set neighbor 192.168.6.7
```

3. Configure the multihop statement to enable Device C and Device E to become EBGP peers.

Because the peers are two hops away from each other, the example uses the `ttl 2` statement.

```
[edit protocols bgp group external-peers]
user@C# set multihop ttl 2
```

4. Configure connectivity to Device E, using static routes.

   You must configure a route to both the loopback interface address and to the address on the physical interface.

```
[edit  routing-options]
user@C# set static route 10.10.10.14/32 next-hop 10.10.10.10
user@C# set static route 192.168.6.7/32 next-hop 10.10.10.10
```

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

```
[edit routing-options]
user@C# set router-id 192.168.40.4
user@C# set autonomous-system 17
```

6. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-static term 1]
user@C# set from protocol static
user@C# set then accept
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@C# show interfaces
fe-1/2/0 {
    unit 9 {
```

```
        description to-D;
        family inet {
            address 10.10.10.9/30;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.40.4/32;
        }
    }
}
```

```
user@C# show protocols
bgp {
    group external-peers {
        type external;
        multihop {
            ttl 2;
        }
        local-address 192.168.40.4;
        export send-static;
        peer-as 18;
        neighbor 192.168.6.7;
    }
}
```

```
user@C# show policy-options
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@C# show routing-options
static {
    route 10.10.10.14/32 next-hop 10.10.10.10;
```

```
    route 192.168.6.7/32 next-hop 10.10.10.10;
}
router-id 192.168.40.4;
autonomous-system 17;
```

If you are done configuring the device, enter `commit` from configuration mode.
Repeat these steps for all BGP sessions in the topology.

Configuring Device D

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 Junos OS CLI User Guide.

To configure Device D:

1. Set the CLI to Device D.

```
user@host> set cli logical-system D
```

2. Configure the interfaces to the directly connected devices, and configure a loopback interface.

```
[edit interfaces fe-1/2/0 unit 10]
user@D# set description to-C
user@D# set family inet address 10.10.10.10/30
[edit interfaces fe-1/2/1 unit 13]
user@D# set description to-E
user@D# set family inet address 10.10.10.13/30
[edit interfaces lo0 unit 4]
user@D# set family inet address 192.168.6.6/32
```

3. Configure connectivity to the other devices using static routes to the loopback interface addresses.

On Device D, you do not need static routes to the physical addresses because Device D is directly connected to Device C and Device E.

```
[edit routing-options]
user@D# set static route 192.168.40.4/32 next-hop 10.10.10.9
user@D# set static route 192.168.6.7/32 next-hop 10.10.10.14
```

4. Configure the local router ID.

```
[edit routing-options]
user@D# set router-id 192.168.6.6
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` 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@D# show interfaces
fe-1/2/0 {
    unit 10 {
        description to-C;
        family inet {
            address 10.10.10.10/30;
        }
    }
}
fe-1/2/1 {
    unit 13 {
        description to-E;
        family inet {
            address 10.10.10.13/30;
        }
    }
}
lo0 {
    unit 4 {
        family inet {
            address 192.168.6.6/32;
```

```
        }
    }
}
```

```
user@D# show protocols
```

```
user@D# show routing-options
static {
    route 192.168.40.4/32 next-hop 10.10.10.9;
    route 192.168.6.7/32 next-hop 10.10.10.14;
}
router-id 192.168.6.6;
```

If you are done configuring the device, enter `commit` from configuration mode.
Repeat these steps for all BGP sessions in the topology.

Configuring Device E

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 Junos OS CLI User Guide.

To configure Device E:

1. Set the CLI to Device E.

```
user@host> set cli logical-system E
```

2. Configure the interface to the directly connected device (to-D), and configure the loopback interface.

```
[edit interfaces fe-1/2/0 unit 14]
user@E# set description to-D
user@E# set family inet address 10.10.10.14/30
[edit interfaces lo0 unit 5]
user@E# set family inet address 192.168.6.7/32
```

3. Configure an EBGP session with Device E.

   The `neighbor` statement points to the loopback interface on Device C.

```
[edit protocols bgp group external-peers]
user@E# set local-address 192.168.6.7
user@E# set export send-static
user@E# set peer-as 17
user@E# set neighbor 192.168.40.4
```

4. Configure the `multihop` statement to enable Device C and Device E to become EBGP peers.

   Because the peers are two hops away from each other, the example uses the `ttl 2` statement.

```
[edit protocols bgp group external-peers]
user@E# set multihop ttl 2
```

5. Configure connectivity to Device E, using static routes.

   You must configure a route to both the loopback interface address and to the address on the physical interface.

```
[edit routing-options]
user@E# set static route 10.10.10.8/30 next-hop 10.10.10.13
user@E# set static route 192.168.40.4/32 next-hop 10.10.10.13
```

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

```
[edit routing-options]
user@E# set router-id 192.168.6.7
user@E# set autonomous-system 18
```

7. Configure a policy that accepts direct routes.

Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-static term 1]
user@E# set from protocol static
user@E# set then accept
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@E# show interfaces
fe-1/2/0 {
    unit 14 {
        description to-D;
        family inet {
            address 10.10.10.14/30;
        }
    }
}
lo0 {
    unit 5 {
        family inet {
            address 192.168.6.7/32;
        }
    }
}
```

```
user@E# show protocols
bgp {
    group external-peers {
        multihop {
            ttl 2;
        }
        local-address 192.168.6.7;
        export send-static;
        peer-as 17;
```

```
        neighbor 192.168.40.4;
    }
}
```

```
user@E# show policy-options
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@E# show routing-options
static {
    route 10.10.10.8/30 next-hop 10.10.10.13;
    route 192.168.40.4/32 next-hop 10.10.10.13;
}
router-id 192.168.6.7;
autonomous-system 18;
```

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

Verification

IN THIS SECTION

- Verifying Connectivity | **426**
- Verifying That BGP Sessions Are Established | **427**
- Viewing Advertised Routes | **427**

Confirm that the configuration is working properly.

Verifying Connectivity

Purpose

Make sure that Device C can ping Device E, specifying the loopback interface address as the source of the ping request.

The loopback interface address is the source address that BGP will use.

Action

From operational mode, enter the `ping 10.10.10.14 source 192.168.40.4` command from Device C, and enter the `ping 10.10.10.9 source 192.168.6.7` command from Device E.

```
user@C> ping 10.10.10.14 source 192.168.40.4

PING 10.10.10.14 (10.10.10.14): 56 data bytes
64 bytes from 10.10.10.14: icmp_seq=0 ttl=63 time=1.262 ms
64 bytes from 10.10.10.14: icmp_seq=1 ttl=63 time=1.202 ms
^C
--- 10.10.10.14 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.202/1.232/1.262/0.030 ms
```

```
user@E> ping 10.10.10.9 source 192.168.6.7

PING 10.10.10.9 (10.10.10.9): 56 data bytes
64 bytes from 10.10.10.9: icmp_seq=0 ttl=63 time=1.255 ms
64 bytes from 10.10.10.9: icmp_seq=1 ttl=63 time=1.158 ms
^C
--- 10.10.10.9 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.158/1.206/1.255/0.049 ms
```

Meaning

The static routes are working if the pings work.

Verifying That BGP Sessions Are Established

Purpose

Verify that the BGP sessions are up.

Action

From operational mode, enter the `show bgp summary` command.

```
user@C> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 2          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.168.6.7              18        147        147       0       1     1:04:27
0/2/2/0              0/0/0/0
```

```
user@E> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 2          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.168.40.4             17        202        202       0       1     1:02:18
0/2/2/0              0/0/0/0
```

Meaning

The output shows that both devices have one peer each. No peers are down.

Viewing Advertised Routes

Purpose

Check to make sure that routes are being advertised by BGP.

Action

From operational mode, enter the `show route advertising-protocol bgp` *neighbor* command.

```
user@E> show route advertising-protocol bgp 192.168.6.7

inet.0: 5 destinations, 7 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.10.10.14/32          Self                                    I
* 192.168.6.7/32          Self                                    I
```

```
user@C> show route advertising-protocol bgp 192.168.40.4

inet.0: 5 destinations, 7 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.10.10.8/30           Self                                    I
* 192.168.40.4/32         Self                                    I
```

Meaning

The `send-static` routing policy is exporting the static routes from the routing table into BGP. BGP is advertising these routes between the peers because the BGP peer session is established.

SEE ALSO

- Understanding EBGP Multihop | 412
- Routing Policies, Firewall Filters, and Traffic Policers User Guide
- Understanding External BGP Peering Sessions | 26
- BGP Configuration Overview | 24

4

CHAPTER

Configuring BGP Session Policies

IN THIS CHAPTER

- Basic BGP Routing Policies | **430**
- Routing Policies for BGP Communities | **509**

Basic BGP Routing Policies

IN THIS SECTION

- Understanding Routing Policies | 430
- Example: Applying Routing Policies at Different Levels of the BGP Hierarchy | 431
- Example: Injecting OSPF Routes into the BGP Routing Table | 444
- Configuring Routing Policies to Control BGP Route Advertisements | 449
- Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers | 455
- Example: Configuring BGP Prefix-Based Outbound Route Filtering | 467
- Understanding the Default BGP Routing Policy on Packet Transport Routers (PTX Series) | 473
- Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers | 476
- Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases | 480
- Conditional Advertisement and Import Policy (Routing Table) with certain match conditions | 482
- Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table | 484
- Implicit filter for Default EBGP Route Propagation Behavior without Policies | 508

Understanding Routing Policies

Each routing policy is identified by a policy name. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in double quotation marks. Each routing policy name must be unique within a configuration.

Once a policy is created and named, it must be applied before it is active. You apply routing policies using the `import` and `export` statements at the `protocols` *`protocol-name`* level in the configuration hierarchy.

In the `import` statement, you list the name of the routing policy to be evaluated when routes are imported into the routing table from the routing protocol.

In the `export` statement, you list the name of the routing policy to be evaluated when routes are being exported from the routing table into a dynamic routing protocol. Only active routes are exported from the routing table.

To specify more than one policy and create a policy chain, you list the policies using a space as a separator. If multiple policies are specified, the policies are evaluated in the order in which they are specified. As soon as an accept or reject action is executed, the policy chain evaluation ends.

SEE ALSO

Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS | 512

Example: Configuring a Global Policy with No Zone Restrictions

Example: Applying Routing Policies at Different Levels of the BGP Hierarchy

IN THIS SECTION

- Requirements | 431
- Overview | 431
- Configuration | 434
- Verification | 440

This example shows BGP configured in a simple network topology and explains how routing polices take effect when they are applied at different levels of the BGP configuration.

Requirements

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

Overview

IN THIS SECTION

- Topology | 433

For BGP, you can apply policies as follows:

- BGP global `import` and `export` statements—Include these statements at the `[edit protocols bgp]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp]` hierarchy level).
- Group `import` and `export` statements—Include these statements at the `[edit protocols bgp group` *`group-name`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`*`]` hierarchy level).
- Peer `import` and `export` statements—Include these statements at the `[edit protocols bgp group` *`group-name`* `neighbor` *`address`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`* `neighbor` *`address`*`]` hierarchy level).
- family `import` and `export` statements—Include these statements at the `[edit protocols bgp family` *`nlri`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp family` *`nlri`*`]` hierarchy level).

A peer-level `import` or `export` statement overrides a group `import` or `export` statement. A group-level `import` or `export` statement overrides a global BGP `import` or `export` statement.

In this example, a policy named `send-direct` is applied at the global level, another policy named `send-192.168.0.1` is applied at the group level, and a third policy named `send-192.168.20.1` is applied at the neighbor level.

```
user@host# show protocols
bgp {
    local-address 172.16.1.1;
    export send-direct;
    group internal-peers {
        type internal;
        export send-192.168.0.1;
        neighbor 172.16.2.2 {
            export send-192.168.20.1;
        }
        neighbor 172.16.3.3;
    }
    group other-group {
        type internal;
        neighbor 172.16.4.4;
    }
}
```

A key point, and one that is often misunderstood and that can lead to problems, is that in such a configuration, only the most explicit policy is applied. A neighbor-level policy is more explicit than a group-level policy, which in turn is more explicit than a global policy.

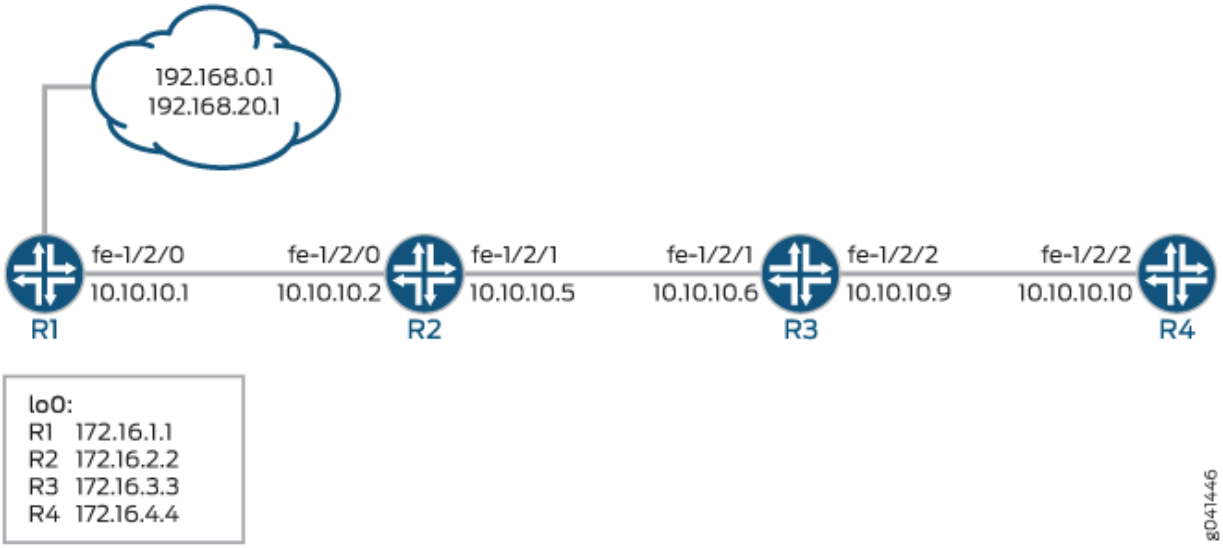
The neighbor 172.16.2.2 is subjected only to the send-192.168.20.1 policy. The neighbor 172.16.3.3, lacking anything more specific, is subjected only to the send-192.168.0.1 policy. Meanwhile, neighbor 172.16.4.4 in group other-group has no group or neighbor-level policy, so it uses the send-direct policy.

If you need to have neighbor 172.16.2.2 perform the function of all three policies, you can write and apply a new neighbor-level policy that encompasses the functions of the other three, or you can apply all three existing policies, as a chain, to neighbor 172.16.2.2.

Topology

Figure 34 on page 433 shows the sample network.

Figure 34: Applying Routing Policies to BGP



"CLI Quick Configuration" on page 434 shows the configuration for all of the devices in Figure 34 on page 433.

The section "No Link Title" on page 436 describes the steps on Device R1.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 434
- Procedure | 436
- Results | 438

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 R1

```
set interfaces fe-1/2/0 unit 0 description to-R2
set interfaces fe-1/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set protocols bgp local-address 172.16.1.1
set protocols bgp export send-direct
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers export send-static-192.168.0
set protocols bgp group internal-peers neighbor 172.16.2.2 export send-static-192.168.20
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols bgp group other-group type internal
set protocols bgp group other-group neighbor 172.16.4.4
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 policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static-192.168.0 term 1 from protocol static
set policy-options policy-statement send-static-192.168.0 term 1 from route-filter
192.168.0.0/24 orlonger
set policy-options policy-statement send-static-192.168.0 term 1 then accept
set policy-options policy-statement send-static-192.168.20 term 1 from protocol static
set policy-options policy-statement send-static-192.168.20 term 1 from route-filter
192.168.20.0/24 orlonger
set policy-options policy-statement send-static-192.168.20 term 1 then accept
```

```
set routing-options static route 192.168.0.1/32 discard
set routing-options static route 192.168.20.1/32 discard
set routing-options router-id 172.16.1.1
set routing-options autonomous-system 17
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.10.10.2/30
set interfaces fe-1/2/1 unit 0 description to-R3
set interfaces fe-1/2/1 unit 0 family inet address 10.10.10.5/30
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.4.4
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 ospf area 0.0.0.0 interface fe-1/2/1.0
set routing-options router-id 172.16.2.2
set routing-options autonomous-system 17
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to-R2
set interfaces fe-1/2/1 unit 0 family inet address 10.10.10.6/30
set interfaces fe-1/2/2 unit 0 description to-R4
set interfaces fe-1/2/2 unit 0 family inet address 10.10.10.9/30
set interfaces lo0 unit 0 family inet address 172.16.3.3/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.3.3
set protocols bgp group internal-peers neighbor 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.4.4
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
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set routing-options router-id 172.16.3.3
set routing-options autonomous-system 17
```

Device R4

```
set interfaces fe-1/2/2 unit 0 description to-R3
set interfaces fe-1/2/2 unit 0 family inet address 10.10.10.10/30
set interfaces lo0 unit 0 family inet address 172.16.4.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.4.4
set protocols bgp group internal-peers neighbor 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set routing-options router-id 172.16.4.4
set routing-options autonomous-system 17
```

Procedure

Step-by-Step Procedure

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

To configure an IS-IS default route policy:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 description to-R2
user@R1# set fe-1/2/0 unit 0 family inet address 10.10.10.1/30
user@R1# set lo0 unit 0 family inet address 172.16.1.1/32
```

2. Enable OSPF, or another interior gateway protocols (IGP), on the interfaces.

```
[edit protocols OSPF area 0.0.0.0]
user@R1# set interface lo0.0 passive
user@R1# set interface fe-1/2/0.0
```

3. Configure static routes.

```
[edit routing-options]
user@R1# set static route 192.168.0.1/32 discard
user@R1# set static route 192.168.20.1/32 discard
```

4. Enable the routing policies.

```
[edit protocols policy-options]
user@R1# set policy-statement send-direct term 1 from protocol direct
user@R1# set policy-statement send-direct term 1 then accept
user@R1# set policy-statement send-static-192.168.0 term 1 from protocol static
user@R1# set policy-statement send-static-192.168.0 term 1 from route-filter 192.168.0.0/24
orlonger
user@R1# set policy-statement send-static-192.168.0 term 1 then accept
user@R1# set policy-statement send-static-192.168.20 term 1 from protocol static
user@R1# set policy-statement send-static-192.168.20 term 1 from route-filter 192.168.20.0/24
orlonger
user@R1# set policy-statement send-static-192.168.20 term 1 then accept
```

5. Configure BGP and apply the export policies.

```
[edit protocols bgp]
user@R1# set local-address 172.16.1.1
user@R1# set protocols bgp export send-direct
user@R1# set group internal-peers type internal
user@R1# set group internal-peers export send-static-192.168.0
user@R1# set group internal-peers neighbor 172.16.2.2 export send-static-192.168.20
user@R1# set group internal-peers neighbor 172.16.3.3
user@R1# set group other-group type internal
user@R1# set group other-group neighbor 172.16.4.4
```

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

```
[edit routing-options]
user@R1# set router-id 172.16.1.1
user@R1# set autonomous-system 17
```

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

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

Results

From configuration mode, confirm your configuration by issuing the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
fe-1/2/0 {
    unit 0 {
        description to-R2;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    local-address 172.16.1.1;
    export send-direct;
    group internal-peers {
        type internal;
        export send-static-192.168.0;
        neighbor 172.16.2.2 {
            export send-static-192.168.20;
        }
        neighbor 172.16.3.3;
```

```
    }
    group other-group {
        type internal;
        neighbor 172.16.4.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface fe-1/2/0.0;
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement send-static-192.168.0 {
    term 1 {
        from {
            protocol static;
            route-filter 192.168.0.0/24 orlonger;
        }
        then accept;
    }
}
policy-statement send-static-192.168.20 {
    term 1 {
        from {
            protocol static;
            route-filter 192.168.20.0/24 orlonger;
        }
        then accept;
```

```
    }
}
```

```
user@R1# show routing-options
static {
    route 192.168.0.1/32 discard;
    route 192.168.20.1/32 discard;
}
router-id 172.16.1.1;
autonomous-system 17;
```

Verification

IN THIS SECTION

- Verifying BGP Route Learning | 440
- Verifying BGP Route Receiving | 442

Confirm that the configuration is working properly.

Verifying BGP Route Learning

Purpose

Make sure that the BGP export policies are working as expected by checking the routing tables.

Action

```
user@R1> show route protocol direct

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.1/32         *[Direct/0] 1d 22:19:47
                    > via lo0.0
```

```
10.10.10.0/30      *[Direct/0] 1d 22:19:47
                    > via fe-1/2/0.0
```

```
user@R1> show route protocol static

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.1/32     *[Static/5] 02:20:03
                      Discard
192.168.20.1/32    *[Static/5] 02:20:03
                      Discard
```

```
user@R2> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.20.1/32    *[BGP/170] 02:02:40, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.1 via fe-1/2/0.0
```

```
user@R3> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.1/32     *[BGP/170] 02:02:51, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.5 via fe-1/2/1.0
```

```
user@R4> show route protocol bgp
inet.0: 9 destinations, 11 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.1/32          [BGP/170] 1d 20:38:54, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.9 via fe-1/2/2.0
10.10.10.0/30       [BGP/170] 1d 20:38:54, localpref 100, from 172.16.1.1
```

```
                    AS path: I, validation-state: unverified
                  > to 10.10.10.9 via fe-1/2/2.0
```

Meaning

On Device R1, the `show route protocol direct` command displays two direct routes: 172.16.1.1/32 and 10.10.10.0/30. The `show route protocol static` command displays two static routes: 192.168.0.1/32 and 192.168.20.1/32.

On Device R2, the `show route protocol bgp` command shows that the only route that Device R2 has learned through BGP is the 192.168.20.1/32 route.

On Device R3, the `show route protocol bgp` command shows that the only route that Device R3 has learned through BGP is the 192.168.0.1/32 route.

On Device R4, the `show route protocol bgp` command shows that the only routes that Device R4 has learned through BGP are the 172.16.1.1/32 and 10.10.10.0/30 routes.

Verifying BGP Route Receiving

Purpose

Make sure that the BGP export policies are working as expected by checking the BGP routes received from Device R1.

Action

```
user@R2> show route receive-protocol bgp 172.16.1.1

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.20.1/32         172.16.1.1                      100        I
```

```
user@R3> show route receive-protocol bgp 172.16.1.1

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

```
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.0.1/32          172.16.1.1                      100        I
```

```
user@R4> show route receive-protocol bgp 172.16.1.1

inet.0: 9 destinations, 11 routes (9 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.1.1/32              172.16.1.1                      100        I
  10.10.10.0/30           172.16.1.1                      100        I
```

Meaning

On Device R2, the `route receive-protocol bgp 172.16.1.1` command shows that Device R2 received only one BGP route, 192.168.20.1/32, from Device R1.

On Device R3, the `route receive-protocol bgp 172.16.1.1` command shows that Device R3 received only one BGP route, 192.168.0.1/32, from Device R1.

On Device R4, the `route receive-protocol bgp 172.16.1.1` command shows that Device R4 received two BGP routes, 172.16.1.1/32 and 10.10.10.0/30, from Device R1.

In summary, when multiple policies are applied at different CLI hierarchies in BGP, only the most specific application is evaluated, to the exclusion of other, less specific policy applications. Although this point might seem to make sense, it is easily forgotten during router configuration, when you mistakenly believe that a neighbor-level policy is combined with a global or group-level policy, only to find that your policy behavior is not as anticipated.

SEE ALSO

> *Example: Configuring Policy Chains and Route Filters*
>
> *Example: Configuring a Policy Subroutine*
>
> *Example: Configuring Routing Policy Prefix Lists*
>
> *export*
>
> *import*

Example: Injecting OSPF Routes into the BGP Routing Table

IN THIS SECTION

- Requirements | 444
- Overview | 444
- Configuration | 445
- Verification | 448
- Troubleshooting | 449

This example shows how to create a policy that injects OSPF routes into the BGP routing table.

Requirements

Before you begin:

- Configure network interfaces.
- Configure external peer sessions. See Example: Configuring External BGP Point-to-Point Peer Sessions.
- Configure interior gateway protocol (IGP) sessions between peers.

Overview

IN THIS SECTION

- Topology | 444

In this example, you create a routing policy called `injectpolicy1` and a routing term called `injectterm1`. The policy injects OSPF routes into the BGP routing table.

Topology

Configuration

IN THIS SECTION

- Configuring the Routing Policy | 445
- Configuring Tracing for the Routing Policy | 447

Configuring the Routing Policy

CLI Quick Configuration

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

```
set policy-options policy-statement injectpolicy1 term injectterm1 from protocol ospf
set policy-options policy-statement injectpolicy1 term injectterm1 from area 0.0.0.1
set policy-options policy-statement injectpolicy1 term injectterm1 then accept
set protocols bgp export injectpolicy1
```

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 inject OSPF routes into a BGP routing table:

1. Create the policy term.

```
[edit policy-options policy-statement injectpolicy1]
user@host# set term injectterm1
```

2. Specify OSPF as a match condition.

```
[edit policy-options policy-statement injectpolicy1 term injectterm1]
user@host# set from protocol ospf
```

3. Specify the routes from an OSPF area as a match condition.

```
[edit policy-options policy-statement injectpolicy1 term injectterm1]
user@host# set from area 0.0.0.1
```

4. Specify that the route is to be accepted if the previous conditions are matched.

```
[edit policy-options policy-statement injectpolicy1 term injectterm1]
user@host# set then accept
```

5. Apply the routing policy to BGP.

```
[edit]
user@host# set protocols bgp export injectpolicy1
```

Results

Confirm your configuration by entering the `show policy-options` and `show protocols bgp` commands from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show policy-options
policy-statement injectpolicy1 {
    term injectterm1 {
        from {
            protocol ospf;
            area 0.0.0.1;
        }
        then accept;
```

```
    }
}
```

```
user@host# show protocols bgp
export injectpolicy1;
```

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

Configuring Tracing for the Routing Policy

CLI Quick Configuration

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

```
set policy-options policy-statement injectpolicy1 term injectterm1 then trace
set routing-options traceoptions file ospf-bgp-policy-log
set routing-options traceoptions file size 5m
set routing-options traceoptions file files 5
set routing-options traceoptions flag policy
```

Step-by-Step Procedure

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

1. Include a trace action in the policy.

```
[edit policy-options policy-statement injectpolicy1 term injectterm1]
user@host# then trace
```

2. Configure the tracing file for the output.

```
[edit routing-options traceoptions]
user@host# set file ospf-bgp-policy-log
user@host# set file size 5m
```

```
user@host# set file files 5
user@host# set flag policy
```

Results

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

```
user@host# show policy-options
policy-statement injectpolicy1 {
    term injectterm1 {
        then {
            trace;
        }
    }
}
```

```
user@host# show routing-options
traceoptions {
    file ospf-bgp-policy-log size 5m files 5;
    flag policy;
}
```

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

Verification

IN THIS SECTION

- Verifying That the Expected BGP Routes Are Present | 449

Confirm that the configuration is working properly.

Verifying That the Expected BGP Routes Are Present

Purpose

Verify the effect of the export policy.

Action

From operational mode, enter the `show route` command.

Troubleshooting

IN THIS SECTION

- Using the show log Command to Examine the Actions of the Routing Policy | 449

Using the show log Command to Examine the Actions of the Routing Policy

Problem

The routing table contains unexpected routes, or routes are missing from the routing table.

Solution

If you configure policy tracing as shown in this example, you can run the `show log ospf-bgp-policy-log` command to diagnose problems with the routing policy. The `show log ospf-bgp-policy-log` command displays information about the routes that the `injectpolicy1` policy term analyzes and acts upon.

Configuring Routing Policies to Control BGP Route Advertisements

IN THIS SECTION

- Applying Routing Policy | 450
- Setting BGP to Advertise Inactive Routes | 451

- Configuring BGP to Advertise the Best External Route to Internal Peers | **452**
- Configuring How Often BGP Exchanges Routes with the Routing Table | **453**
- Disabling Suppression of Route Advertisements | **454**

All routing protocols use the Junos OS routing table to store the routes that they learn and to determine which routes they should advertise in their protocol packets. Routing policy allows you to control which routes the routing protocols store in and retrieve from the routing table. For information about routing policy, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

When configuring BGP routing policy, you can perform the following tasks:

Applying Routing Policy

You define routing policy at the `[edit policy-options]` hierarchy level. To apply policies you have defined for BGP, include the `import` and `export` statements within the BGP configuration.

You can apply policies as follows:

- BGP global `import` and `export` statements—Include these statements at the `[edit protocols bgp]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp]` hierarchy level).
- Group `import` and `export` statements—Include these statements at the `[edit protocols bgp group` *`group-name`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`*`]` hierarchy level).
- Peer `import` and `export` statements—Include these statements at the `[edit protocols bgp group` *`group-name`* `neighbor` *`address`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`* `neighbor` *`address`*`]` hierarchy level).
- family `import` and `export` statements—Include these statements at the `[edit protocols bgp family` *`nlri`*`]` hierarchy level (for routing instances, include these statements at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp family` *`nlri`*`]` hierarchy level).

A peer-level `import` or `export` statement overrides a group `import` or `export` statement. A group-level `import` or `export` statement overrides a global BGP `import` or `export` statement.

To apply policies, see the following sections:

Applying Policies to Routes Being Imported into the Routing Table from BGP

To apply policy to routes being imported into the routing table from BGP, include the `import` statement, listing the names of one or more policies to be evaluated:

```
import [ policy-names ];
```

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

If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching filter is applied to the route. If no match is found, BGP places into the routing table only those routes that were learned from BGP routing devices.

Applying Policies to Routes Being Exported from the Routing Table into BGP

To apply policy to routes being exported from the routing table into BGP, include the `export` statement, listing the names of one or more policies to be evaluated:

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

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

If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching filter is applied to the route. If no routes match the filters, the routing table exports into BGP only the routes that it learned from BGP.

Setting BGP to Advertise Inactive Routes

By default, BGP stores the route information it receives from update messages in the Junos OS routing table, and the routing table exports only active routes into BGP, which BGP then advertises to its peers. To have the routing table export to BGP the best route learned by BGP even if Junos OS did not select it to be an active route, include the `advertise-inactive` statement:

```
advertise-inactive;
```

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

Configuring BGP to Advertise the Best External Route to Internal Peers

In general, deployed BGP implementations do not advertise the external route with the highest local preference value to internal peers unless it is the best route. Although this behavior was required by an earlier version of the BGP version 4 specification, RFC 1771, it was typically not followed in order to minimize the amount of advertised information and to prevent routing loops. However, there are scenarios in which advertising the best external route is beneficial, in particular, situations that can result in IBGP route oscillation.

In Junos OS Release 9.3 and later, you can configure BGP to advertise the best external route into an internal BGP (IBGP) mesh group, a route reflector cluster, or an autonomous system (AS) confederation, even when the best route is an internal route.

NOTE: In order to configure the `advertise-external` statement on a route reflector, you must disable intracluster reflection with the `no-client-reflect` statement.

When a routing device is configured as a route reflector for a cluster, a route advertised by the route reflector is considered internal if it is received from an internal peer with the same cluster identifier or if both peers have no cluster identifier configured. A route received from an internal peer that belongs to another cluster, that is, with a different cluster identifier, is considered external.

In a confederation, when advertising a route to a confederation border router, any route from a different confederation sub-AS is considered external.

You can also configure BGP to advertise the external route only if the route selection process reaches the point where the multiple exit discriminator (MED) metric is evaluated. As a result, an external route with an AS path worse (that is, longer) than that of the active path is not advertised.

Junos OS also provides support for configuring a BGP export policy that matches on the state of an advertised route. You can match on either active or inactive routes. For more information, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

To configure BGP to advertise the best external path to internal peers, include the `advertise-external` statement:

```
advertise-external;
```

NOTE: The `advertise-external` statement is supported at both the group and neighbor level. If you configure the statement at the neighbor level, you must configure it for all neighbors in a group. Otherwise, the group is automatically split into different groups.

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

To configure BGP to advertise the best external path only if the route selection process reaches the point where the MED value is evaluated, include the `conditional` statement:

```
advertise-external {
    conditional;
}
```

Configuring How Often BGP Exchanges Routes with the Routing Table

BGP stores the route information it receives from update messages in the routing table, and the routing table exports active routes from the routing table into BGP. BGP then advertises the exported routes to its peers. By default, the exchange of route information between BGP and the routing table occurs immediately after the routes are received. This immediate exchange of route information might cause instabilities in the network reachability information. To guard against this, you can delay the time between when BGP and the routing table exchange route information.

To configure how often BGP and the routing table exchange route information, include the `out-delay` statement:

```
out-delay seconds;
```

By default, the routing table retains some of the route information learned from BGP. To have the routing table retain all or none of this information, include the `keep` statement:

```
keep (all | none);
```

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

The routing table can retain the route information learned from BGP in one of the following ways:

- Default (omit the `keep` statement)—Keep all route information that was learned from BGP, except for routes whose AS path is looped and whose loop includes the local AS.
- `keep all`—Keep all route information that was learned from BGP.
- `keep none`—Discard routes that were received from a peer and that were rejected by import policy or other sanity checking, such as AS path or next hop. When you configure `keep none` for the BGP session

and the inbound policy changes, Junos OS forces readvertisement of the full set of routes advertised by the peer.

In an AS path healing situation, routes with looped paths theoretically could become usable during a soft reconfiguration when the AS path loop limit is changed. However, there is a significant memory usage difference between the default and `keep all`.

Consider the following scenarios:

- A peer readvertises routes back to the peer from which it learned them.

  This can happen in the following cases:

  - Another vendor's routing device advertises the routes back to the sending peer.
  - The Junos OS peer's default behavior of not readvertising routes back to the sending peer is overridden by configuring `advertise-peer-as`.

- A provider edge (PE) routing device discards any VPN route that does not have any of the expected route targets.

When `keep all` is configured, the behavior of discarding routes received in the above scenarios is overridden.

Disabling Suppression of Route Advertisements

Junos OS does not advertise the routes learned from one EBGP peer back to the same external BGP (EBGP) peer. In addition, the software does not advertise those routes back to any EBGP peers that are in the same AS as the originating peer, regardless of the routing instance. You can modify this behavior by including the `advertise-peer-as` statement in the configuration. To disable the default advertisement suppression, include the `advertise-peer-as` statement:

```
advertise-peer-as;
```

NOTE: The route suppression default behavior is disabled if the `as-override` statement is included in the configuration.

If you include the `advertise-peer-as` statement in the configuration, BGP advertises the route regardless of this check.

To restore the default behavior, include the `no-advertise-peer-as` statement in the configuration:

```
no-advertise-peer-as;
```

If you include both the `as-override` and `no-advertise-peer-as` statements in the configuration, the `no-advertise-peer-as` statement is ignored. You can include these statements at multiple hierarchy levels.

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

SEE ALSO

Example: Configuring BGP Prefix-Based Outbound Route Filtering | 467

Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers

IN THIS SECTION

- Requirements | 457
- Overview | 457
- Configuration | 458
- Verification | 463

The BGP protocol specification, as defined in RFC 1771, specifies that a BGP peer shall advertise to its internal peers the higher preference external path, even if this path is not the overall best (in other words, even if the best path is an internal path). In practice, deployed BGP implementations do not follow this rule. The reasons for deviating from the specification are as follows:

- Minimizing the amount of advertised information. BGP scales according to the number of available paths.
- Avoiding routing and forwarding loops.

There are, however, several scenarios in which the behavior, specified in RFC 1771, of advertising the best external route might be beneficial. Limiting path information is not always desirable as path

diversity might help reduce restoration times. Advertising the best external path can also address internal BGP (IBGP) route oscillation issues as described in RFC 3345, *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*.

The `advertise-external` statement modifies the behavior of a BGP speaker to advertise the best external path to IBGP peers, even when the best overall path is an internal path.

NOTE: The `advertise-external` statement is supported at both the group and neighbor level. If you configure the statement at the neighbor level, you must configure it for all neighbors in a group. Otherwise, the group is automatically split into different groups.

The `conditional` option limits the behavior of the `advertise-external` setting, such that the external route is advertised only if the route selection process reaches the point where the multiple exit discriminator (MED) metric is evaluated. Thus, an external route is not advertised if it has, for instance, an AS path that is worse (longer) than that of the active path. The `conditional` option restricts external path advertisement to when the best external path and the active path are equal until the MED step of the route selection process. Note that the criteria used for selecting the best external path is the same whether or not the `conditional` option is configured.

Junos OS also provides support for configuring a BGP export policy that matches the state of an advertised route. You can match either active or inactive routes, as follows:

```
policy-options {
    policy-statement name{
        from state (active|inactive);
    }
}
```

This qualifier only matches when used in the context of an export policy. When a route is being advertised by a protocol that can advertise inactive routes (such as BGP), `state inactive` matches routes advertised as a result of the `advertise-inactive` and `advertise-external` statements.

For example, the following configuration can be used as a BGP export policy toward internal peers to mark routes advertised due to the `advertise-external` setting with a user-defined community. That community can be later used by the receiving routers to filter out such routes from the forwarding table. Such a mechanism can be used to address concerns that advertising paths not used for forwarding by the sender might lead to forwarding loops.

```
user@host# show policy-options
policy-statement mark-inactive {
    term inactive {
        from state inactive;
```

```
        then {
            community set comm-inactive;
        }
    }
    term default {
        from protocol bgp;
        then accept;
    }
    then reject;
}
community comm-inactive members 65536:65284;
```

Requirements

Junos OS 9.3 or later is required.

Overview

IN THIS SECTION

- Topology | **458**

This example shows three routing devices. Device R2 has an external BGP (EBGP) connection to Device R1. Device R2 has an IBGP connection to Device R3.

Device R1 advertises 172.16.6.0/24. Device R2 does not set the local preference in an import policy for Device R1's routes, and thus 172.16.6.0/24 has the default local preference of 100.

Device R3 advertises 172.16.6.0/24 with a local preference of 200.

When the `advertise-external` statement is not configured on Device R2, 172.16.6.0/24 is not advertised by Device R2 toward Device R3.

When the `advertise-external` statement is configured on Device R2 on the session toward Device R3, 172.16.6.0/24 is advertised by Device R2 toward Device R3.

When `advertise-external conditional` is configured on Device R2 on the session toward Device R3, 172.16.6.0/24 is not advertised by Device R2 toward Device R3. If you remove the `then local-preference 200` setting on Device R3 and add the `path-selection as-path-ignore` setting on Device R2 (thus making the path selection criteria equal until the MED step of the route selection process), 172.16.6.0/24 is advertised by Device R2 toward Device R3.

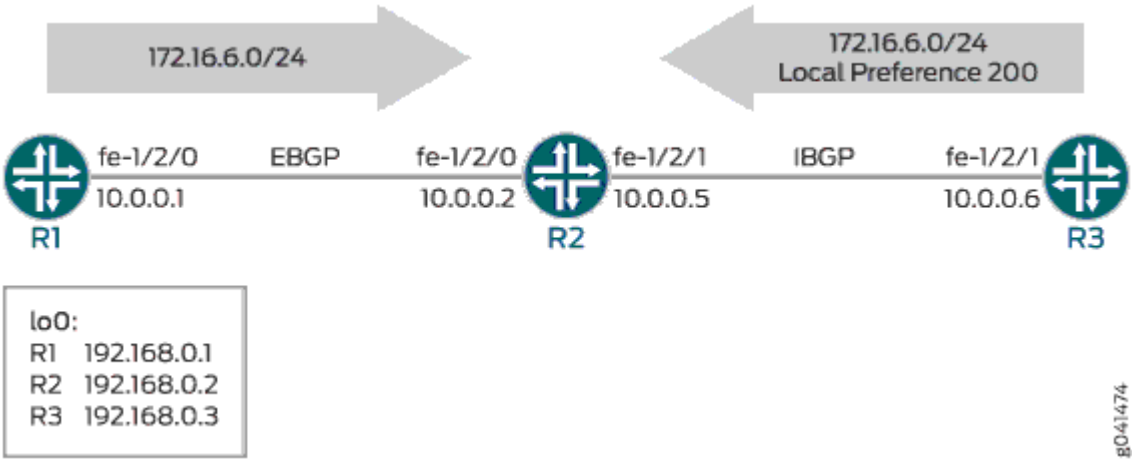
NOTE: To configure the `advertise-external` statement on a route reflector, you must disable intracluster reflection with the `no-client-reflect` statement, and the client cluster must be fully meshed to prevent the sending of redundant route advertisements.

When a routing device is configured as a route reflector for a cluster, a route advertised by the route reflector is considered internal if it is received from an internal peer with the same cluster identifier or if both peers have no cluster identifier configured. A route received from an internal peer that belongs to another cluster, that is, with a different cluster identifier, is considered external.

Topology

Figure 35 on page 458 shows the sample network.

Figure 35: BGP Topology for advertise-external



"CLI Quick Configuration" on page 459 shows the configuration for all of the devices in Figure 35 on page 458.

The section "No Link Title" on page 460 describes the steps on Device R2.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 459
- Procedure | 460

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 R1

```
set interfaces fe-1/2/0 unit 0 description to-R2
set interfaces fe-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 ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 from route-filter 172.16.6.0/24 exact
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 then reject
set routing-options static route 172.16.6.0/24 reject
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 description to-R3
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 100
set protocols bgp group ext neighbor 10.0.0.1
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int advertise-external
set protocols bgp group int neighbor 192.168.0.3
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 routing-options router-id 192.168.0.2
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.6
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then local-preference 200
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.6.0/24 reject
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.5
set routing-options autonomous-system 200
```

Procedure

Step-by-Step Procedure

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

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 description to-R1
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 0 description to-R3
user@R2# set fe-1/2/1 unit 0 family inet address 10.0.0.5/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure OSPF or another interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface fe-1/2/1.0
user@R2# set interface lo0.0 passive
```

3. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set peer-as 100
user@R2# set neighbor 10.0.0.1
```

4. Configure the IBGP connection to Device R3.

```
[edit protocols bgp group int]
user@R2# set type internal
user@R2# set local-address 192.168.0.2
user@R2# set neighbor 192.168.0.3
```

5. Add the `advertise-external` statement to the IBGP group peering session.

```
[edit protocols bgp group int]
user@R2# set advertise-external
```

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

```
[edit routing-options ]
user@R2# set router-id 192.168.0.2
user@R2# set autonomous-system 200
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
fe-1/2/0 {
    unit 0{
        description to-R1;
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        description to-R3;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        type external;
        peer-as 100;
        neighbor 10.0.0.1;
    }
    group int {
        type internal;
        local-address 192.168.0.2;
```

```
        advertise-external;
        neighbor 192.168.0.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R2# show routing-options
router-id 192.168.0.2;
autonomous-system 200;
```

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

Verification

IN THIS SECTION

- Verifying the BGP Active Path | **463**
- Verifying the External Route Advertisement | **464**
- Verifying the Route on Device R3 | **465**
- Experimenting with the conditional Option | **465**

Confirm that the configuration is working properly.

Verifying the BGP Active Path

Purpose

On Device R2, make sure that the 172.16.6.0/24 prefix is in the routing table and has the expected active path.

Action

```
user@R2> show route 172.16.6

inet.0: 8 destinations, 9 routes (8 active, 1 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24      *[BGP/170] 00:00:07, localpref 200, from 192.168.0.3
                      AS path: I, validation-state: unverified
                     > to 10.0.0.6 via fe-1/2/1.0
                     [BGP/170] 03:23:03, localpref 100
                      AS path: 100 I, validation-state: unverified
                     > to 10.0.0.1 via fe-1/2/0.0
```

Meaning

Device R2 receives the 172.16.6.0/24 route from both Device R1 and Device R3. The route from Device R3 is the active path, as designated by the asterisk (*). The active path has the highest local preference. Even if the local preferences of the two routes were equal, the route from Device R3 would remain active because it has the shortest AS path.

Verifying the External Route Advertisement

Purpose

On Device R2, make sure that the 172.16.6.0/24 route is advertised toward Device R3.

Action

```
user@R2> show route advertising-protocol bgp 192.168.0.3

inet.0: 8 destinations, 9 routes (8 active, 1 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.6.0/24           10.0.0.1                     100        100 I
```

Meaning

Device R2 is advertising the 172.16.6.0/24 route toward Device R3.

Verifying the Route on Device R3

Purpose

Make sure that the 172.16.6.0/24 prefix is in Device R3's routing table.

Action

```
user@R3> show route 172.16.6.0/24

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24      *[Static/5] 03:34:14
                      Reject
                    [BGP/170] 06:34:43, localpref 100, from 192.168.0.2
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/0.6
```

Meaning

Device R3 has the static route and the BGP route for 172.16.6.0/24.

Note that the BGP route is hidden on Device R3 if the route is not reachable or if the next hop cannot be resolved. To fulfill this requirement, this example includes a static default route on Device R3 (`static route 0.0.0.0/0 next-hop 10.0.0.5`).

Experimenting with the conditional Option

Purpose

See how the `conditional` option works in the context of the BGP path selection algorithm.

Action

1. On Device R2, add the `conditional` option.

```
[edit  protocols bgp group int]
user@R2# set advertise-external conditional
user@R2# commit
```

2. On Device R2, check to see if the 172.16.6.0/24 route is advertised toward Device R3.

```
user@R2> show route advertising-protocol bgp 192.168.0.3
```

   As expected, the route is no longer advertised. You might need to wait a few seconds to see this result.

3. On Device R3, deactivate the `then local-preference` policy action.

```
[edit policy-options policy-statement send-static term 1]
user@R3# deactivate logical-systems R3 then local-preference
user@R3# commit
```

4. On Device R2, ensure that the local preferences of the two paths are equal.

```
user@R2> show route 172.16.6.0/24

inet.0: 8 destinations, 9 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24      *[BGP/170] 08:02:59, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 00:07:51, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/1.0
```

5. On Device R2, add the `as-path-ignore` statement.

```
[edit protocols bgp]
user@R2# set path-selection as-path-ignore
user@R2# commit
```

6. On Device R2, check to see if the 172.16.6.0/24 route is advertised toward Device R3.

```
user@R2> show route advertising-protocol bgp 192.168.0.3

inet.0: 8 destinations, 9 routes (8 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.6.0/24           10.0.0.1                     100        100 I
```

As expected, the route is now advertised because the AS path length is ignored and because the local preferences are equal.

SEE ALSO

Example: Configuring BGP to Advertise Inactive Routes | 316

Understanding BGP Path Selection | 12

Example: Configuring BGP Prefix-Based Outbound Route Filtering

IN THIS SECTION

- Requirements | 468
- Overview | 468
- Configuration | 469
- Verification | 471

This example shows how to configure a Juniper Networks router to accept route filters from remote peers and perform outbound route filtering using the received filters.

Requirements

Before you begin:

- Configure the router interfaces.
- Configure an interior gateway protocol (IGP).

Overview

IN THIS SECTION

- Topology | 468

You can configure a BGP peer to accept route filters from remote peers and perform outbound route filtering using the received filters. By filtering out unwanted updates, the sending peer saves resources needed to generate and transmit updates, and the receiving peer saves resources needed to process updates. This feature can be useful, for example, in a virtual private network (VPN) in which subsets of customer edge (CE) devices are not capable of processing all the routes in the VPN. The CE devices can use prefix-based outbound route filtering to communicate to the provider edge (PE) routing device to transmit only a subset of routes, such as routes to the main data centers only.

The maximum number of prefix-based outbound route filters that a BGP peer can accept is 5000. If a remote peer sends more than 5000 outbound route filters to a peer address, the additional filters are discarded, and a system log message is generated.

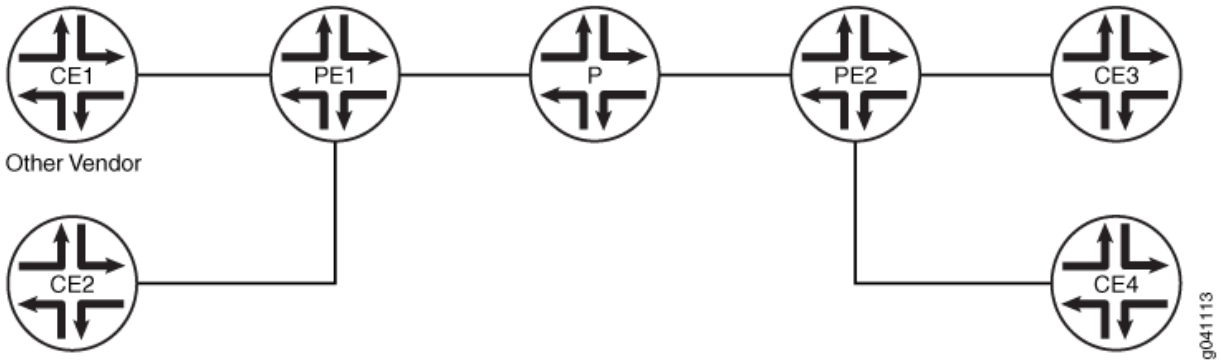
You can configure interoperability for the routing device as a whole or for specific BGP groups or peers only.

Topology

In the sample network, Device CE1 is a router from another vendor. The configuration shown in this example is on Juniper Networks Router PE1.

Figure 36 on page 469 shows the sample network.

Figure 36: BGP Prefix-Based Outbound Route Filtering



Configuration

IN THIS SECTION

- CLI Quick Configuration | 469
- Procedure | 470

CLI Quick Configuration

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

PE1

```
set protocols bgp group cisco-peers type external
set protocols bgp group cisco-peers description “to CE1”
set protocols bgp group cisco-peers local-address 192.168.165.58
set protocols bgp group cisco-peers peer-as 35
set protocols bgp group cisco-peers outbound-route-filter bgp-orf-cisco-mode
set protocols bgp group cisco-peers outbound-route-filter prefix-based accept inet
set protocols bgp group cisco-peers neighbor 192.168.165.56
set routing-options autonomous-system 65500
```

Procedure

Step-by-Step Procedure

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

To configure Router PE1 to accept route filters from Device CE1 and perform outbound route filtering using the received filters:

1. Configure the local autonomous system.

```
[edit routing-options]
user@PE1# set autonomous-system 65500
```

2. Configure external peering with Device CE1.

```
[edit protocols bgp group cisco-peers]
user@PE1# set type external
user@PE1# set description “to CE1”
user@PE1# set local-address 192.168.165.58
user@PE1# set peer-as 35
user@PE1# set neighbor 192.168.165.56
```

3. Configure Router PE1 to accept IPv4 route filters from Device CE1 and perform outbound route filtering using the received filters.

```
[edit protocols bgp group cisco-peers]
user@PE1# set outbound-route-filter prefix-based accept inet
```

4. (Optional) Enable interoperability with routing devices that use the vendor-specific compatibility code of 130 for outbound route filters and the code type of 128.

   The IANA standard code is 3, and the standard code type is 64.

```
[edit protocols bgp group cisco-peers]
user@PE1# set outbound-route-filter bgp-orf-cisco-mode
```

Results

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

```
user@PE1# show protocols
group cisco-peers {
    type external;
    description “to CE1”;
    local-address 192.168.165.58;
    peer-as 35;
    outbound-route-filter {
        bgp-orf-cisco-mode;
        prefix-based {
            accept {
                inet;
            }
        }
    }
    neighbor 192.168.165.56;
}
```

```
user@PE1# show routing-options
autonomous-system 65500;
```

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

Verification

IN THIS SECTION

- Verifying the Outbound Route Filter | **472**
- Verifying the BGP Neighbor Mode | **472**

Confirm that the configuration is working properly.

Verifying the Outbound Route Filter

Purpose

Display information about the prefix-based outbound route filter received from Device CE1.

Action

From operational mode, enter the `show bgp neighbor orf detail` command.

```
user@PE1> show bgp neighbor orf 192.168.165.56 detail
   Peer: 192.168.165.56 Type: External
   Group: cisco-peers

   inet-unicast
    Filter updates recv:          4 Immediate:          0
    Filter: prefix-based            receive
            Updates recv:          4
      Received filter entries:
        seq 10 2.2.0.0/16 deny minlen 0 maxlen 0
        seq 20 3.3.0.0/16 deny minlen 24 maxlen 0
        seq 30 4.4.0.0/16 deny minlen 0 maxlen 28
        seq 40 5.5.0.0/16 deny minlen 24 maxlen 28
```

Verifying the BGP Neighbor Mode

Purpose

Verify that the `bgp-orf-cisco-mode` setting is enabled for the peer by making sure that the `ORFCiscoMode` option is displayed in the `show bgp neighbor` command output.

Action

From operational mode, enter the `show bgp neighbor` command.

```
user@PE1> show bgp neighbor
Peer: 192.168.165.56 AS 35       Local: 192.168.165.58 AS 65500
  Type: External    State: Active         Flags: <>
  Last State: Idle          Last Event: Start
  Last Error: None
```

```
Export: [ adv_stat ]
Options: <Preference LocalAddress AddressFamily PeerAS Refresh>
Options: <ORF ORFCiscoMode>
Address families configured: inet-unicast
Local Address: 192.168.165.58 Holdtime: 90 Preference: 170
Number of flaps: 0
Trace options: detail open detail refresh
Trace file: /var/log/orf size 5242880 files 20
```

SEE ALSO

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

Understanding the Default BGP Routing Policy on Packet Transport Routers (PTX Series)

On PTX Series Packet Transport routers, the default BGP routing policy differs from that of other Junos OS routing devices. The default routing policy of the PTX Series 3000 and 5000 Series routers will not install BGP routes in the forwarding table, unless you configure another policy to override it. All other PTX Series routers will install BGP learned routes to the forwarding information base (FIB) and packet forwading engine (PFE) without the need for a policy.

The PTX Series routers are MPLS transit platforms that do IP forwarding, typically using interior gateway protocol (IGP) routes. The PTX Series Packet Forwarding Engine can accommodate a relatively small number of variable-length prefixes.

NOTE: A PTX Series router can support full BGP routes in the control plane so that it can be used as a route reflector (RR). It can do exact-length lookup multicast forwarding and can build the multicast forwarding plane for use by the unicast control plane (for example. to perform a reverse-path forwarding lookup for multicast).

Given the PFE limitation, the default routing policy for PTX Series routers is for BGP routes not to be installed in the forwarding table. You can override the default routing policy and select certain BGP routes to install in the forwarding table.

The default behavior for load balancing and BGP routes on PTX Series routers is as follows. It has the following desirable characteristics:

- Allows you to override the default behavior without needing to alter the default policy directly
- Reduces the chance of accidental changes that nullify the defaults
- Sets no flow-control actions, such as accept and reject

The default routing policy on the PTX Series routers is as follows:

```
user@host# show policy-options | display inheritance defaults no-comments
policy-options {
    policy-statement junos-ptx-series-default {
        term t1 {
            from {
                protocol bgp;
                rib inet.0;
            }
            then no-install-to-fib;
        }
        term t2 {
            from {
                protocol bgp;
                rib inet6.0;
            }
            then no-install-to-fib;
        }
        term t3 {
            then load-balance per-packet;
        }
    }
}
routing-options {
    forwarding-table {
        default-export junos-ptx-series-default;
    }
}
user@host# show routing-options forwarding-table default-export | display inheritance defaults
no-comments
default-export junos-ptx-series-default;
```

As shown here, the junos-ptx-series-default policy is defined in [edit policy-options]. The policy is applied in [edit routing-options forwarding-table], using the default-export statement. You can view these default configurations by using the | display inheritance flag.

Also, you can use the show policy command to view the default policy.

```
user@host> show policy junos-ptx-series-default
Policy junos-ptx-series-default:
    Term t1:
        from proto BGP
         inet.0
        then install-to-fib no
    Term t2:
        from proto BGP
         inet6.0
        then install-to-fib no
    Term t3:
        then load-balance per-packet
```

CAUTION: We strongly recommend that you do not alter the junos-ptx-series-default routing policy directly.

Junos OS chains the junos-ptx-series-default policy and any user-configured export policy. Because the junos-ptx-series-default policy does not use flow-control actions, any export policy that you configure is executed (by way of the implicit next-policy action) for every route. Thus you can override any actions set by the junos-ptx-series-default policy. If you do not configure an export policy, the actions set by junos-ptx-series-default policy are the only actions.

You can use the policy action install-to-fib to override the no-install-to-fib action.

Similarly, you can set the load-balance per-prefix action to override the load-balance per-packet action.

SEE ALSO

Conditional Advertisement and Import Policy (Routing Table) with certain match conditions | 482

Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers

IN THIS SECTION

- Requirements | 476
- Overview | 476
- Configuration | 477
- Verification | 479

This example shows how to override the default routing policy on packet transport routers, such as the PTX Series Packet Transport Routers.

Requirements

This example requires Junos OS Release 12.1 or later.

Overview

By default, the PTX Series routers do not install BGP routes in the forwarding table.

For PTX Series routers, the configuration of the `from protocols bgp` condition with the `then accept` action does not have the usual result that it has on other Junos OS routing devices. With the following routing policy on PTX Series routers, BGP routes do not get installed in the forwarding table.

```
user@host# show policy-options
policy-statement accept-no-install {
    term 1 {
        from protocol bgp;
        then accept;
    }
}
user@host# show routing-options
forwarding-table {
```

```
    export accept-no-install;
}
```

```
user@host> show route forwarding-table
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     2
```

No BGP routes are installed in the forwarding table. This is the expected behavior.

This example shows how to use the `then install-to-fib` action to effectively override the default BGP routing policy.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **477**
- Installing Selected BGP Routes in the Forwarding Table | **478**

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.

```
set policy-options prefix-list install-bgp 66.0.0.1/32
set policy-options policy-statement override-ptx-series-default term 1 from prefix-list install-
bgp
set policy-options policy-statement override-ptx-series-default term 1 then load-balance per-
prefix
set policy-options policy-statement override-ptx-series-default term 1 then install-to-fib
set routing-options forwarding-table export override-ptx-series-default
```

Installing Selected BGP Routes in the Forwarding Table

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 Junos OS CLI User Guide.

To install selected BGP routes in the forwarding table:

1. Configure a list of prefixes to install in the forwarding table.

```
[edit policy-options prefix-list install-bgp]
user@host# set 66.0.0.1/32
```

2. Configure the routing policy, applying the prefix list as a condition.

```
[edit policy-options policy-statement override-ptx-series-default term 1]
user@host# set from prefix-list install-bgp
user@host# set then install-to-fib
user@host# set then load-balance per-prefix
```

3. Apply the routing policy to the forwarding table.

```
[edit routing-options forwarding-table]
user@host# set export override-ptx-series-default
```

Results

From configuration mode, confirm your configuration by entering the `show policy-options` 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@host# show policy-options
prefix-list install-bgp {
    66.0.0.1/32;
}
policy-statement override-ptx-series-default {
    term 1 {
```

```
        from {
            prefix-list install-bgp;
        }
        then {
            load-balance per-prefix;
            install-to-fib;
        }
    }
}
```

```
user@host# show routing-options
forwarding-table {
    export override-ptx-series-default;
}
```

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

Verification

IN THIS SECTION

- Verifying That the Selected Route Is Installed in the Forwarding Table | 479

Confirm that the configuration is working properly.

Verifying That the Selected Route Is Installed in the Forwarding Table

Purpose

Make sure that the configured policy overrides the default policy.

Action

From operational mode, enter the `show route forwarding-table` command.

```
user@host> show route forwarding-table destination 66.0.0.1
Internet:
```

```
Destination        Type RtRef Next hop           Type Index NhRef Netif
66.0.0.1/32        user     0                    indr 2097159     3
                                                 ulst 2097156     2
                              5.1.0.2            ucst   574     1 et-6/0/0.1
                              5.2.0.2            ucst   575     1 et-6/0/0.2
```

Meaning

This output shows that the route to 66.0.0.1/32 is installed in the forwarding table.

SEE ALSO

Basic BGP Routing Policies | 430

Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases

Networks are usually subdivided into smaller, more-manageable units called autonomous systems (ASs). When BGP is used by routers to form peer relationships in the same AS, it is referred to as internal BGP (IBGP). When BGP is used by routers to form peer relationships in different ASs, it is referred to as external BGP (EBGP).

After performing route sanity checks, a BGP router accepts the routes received from its peers and installs them into the routing table. By default, all routers in IBGP and EBGP sessions follow the standard BGP advertisement rules. While a router in an IBGP session advertises only the routes learned from its direct peers, a router in an EBGP session advertises all routes learned from its direct and indirect peers (peers of peers). Hence, in a typical network configured with EBGP, a router adds all routes received from an EBGP peer into its routing table and advertises nearly all routes to all EBGP peers.

A service provider exchanging BGP routes with both customers and peers on the Internet is at risk of malicious and unintended threats that can compromise the proper routing of traffic, as well as the operation of the routers.

This has several disadvantages:

- **Non-aggregated route advertisements**—A customer could erroneously advertise all its prefixes to the ISP rather than an aggregate of its address space. Given the size of the Internet routing table, this must be carefully controlled. An edge router might also need only a default route out toward the Internet and instead be receiving the entire BGP routing table from its upstream peer.

- **BGP route manipulation**—If a malicious administrator alters the contents of the BGP routing table, it could prevent traffic from reaching its intended destination.
- **BGP route hijacking**—A rogue administrator of a BGP peer could maliciously announce a network's prefixes in an attempt to reroute the traffic intended for the victim network to the administrator's network to either gain access to the contents of traffic or to block the victim's online services.
- **BGP denial of service (DoS)**—If a malicious administrator sends unexpected or undesirable BGP traffic to a router in an attempt to use all of the router's available BGP resources, it might result in impairing the router's ability to process valid BGP route information.

Conditional installation of prefixes can be used to address all the problems previously mentioned. If a customer requires access to remote networks, it is possible to install a specific route in the routing table of the router that is connected with the remote network. This does not happen in a typical EBGP network and hence, conditional installation of prefixes becomes essential.

ASs are not only bound by physical relationships but by business or other organizational relationships. An AS can provide services to another organization, or act as a transit AS between two other ASs. These transit ASs are bound by contractual agreements between the parties that include parameters on how to connect to each other and most importantly, the type and quantity of traffic they carry for each other. Therefore, for both legal and financial reasons, service providers must implement policies that control how BGP routes are exchanged with neighbors, which routes are accepted from those neighbors, and how those routes affect the traffic between the ASs.

There are many different options available to filter routes received from a BGP peer to both enforce inter-AS policies and mitigate the risks of receiving potentially harmful routes. Conventional route filtering examines the attributes of a route and accepts or rejects the route based on such attributes. A policy or filter can examine the contents of the AS-Path, the next-hop value, a community value, a list of prefixes, the address family of the route, and so on.

In some cases, the standard "acceptance condition" of matching a particular attribute value is not enough. The service provider might need to use another condition outside of the route itself, for example, another route in the routing table. As an example, it might be desirable to install a default route received from an upstream peer, only if it can be verified that this peer has reachability to other networks further upstream. This conditional route installation avoids installing a default route that is used to send traffic toward this peer, when the peer might have lost its routes upstream, leading to black-holed traffic. To achieve this, the router can be configured to search for the presence of a particular route in the routing table, and based on this knowledge accept or reject another prefix.

"Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table" on page 484 explains how the conditional installation of prefixes can be configured and verified.

SEE ALSO

Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table | 484

Conditional Advertisement and Import Policy (Routing Table) with certain match conditions

BGP accepts all non-looped routes learned from neighbors and imports them into the RIB-In table. If these routes are accepted by the BGP import policy, they are then imported into the inet.0 routing table. In cases where only certain routes are required to be imported, provisions can be made such that the peer routing device exports routes based on a condition or a set of conditions.

The condition for exporting a route can be based on:

- The peer the route was learned from
- The interface the route was learned on
- Some other required attribute

For example:

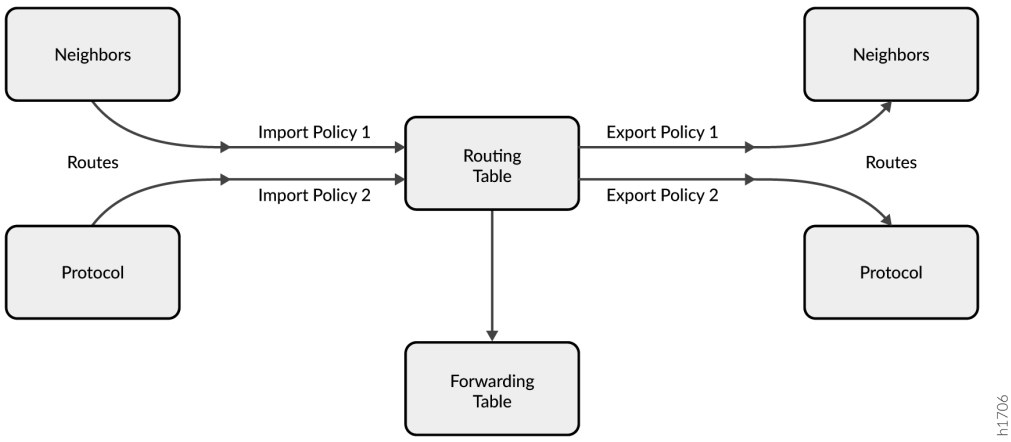
```
[edit]
policy-options {
    condition condition-name {
        if-route-exists address table table-name;
    }
}
```

This is known as conditional installation of prefixes and is described in "Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table" on page 484.

Conditions in routing policies can be configured irrespective of whether they are a part of the export or import policies or both. The export policy supports these conditions inherited from the routing policy based on the existence of another route in the routing policy. However, the import policy doesn't support these conditions, and the conditions are not executed even if they are present.

Figure 37 on page 483 illustrates where BGP import and export policies are applied. An import policy is applied to inbound routes that are visible in the output of the `show route receive-protocol bgp` *`neighbor-address`* command. An export policy is applied to outbound routes that are visible in the output of the `show route advertising-protocol bgp` *`neighbor-address`* command.

Figure 37: BGP Import and Export Policies



To enable conditional installation of prefixes, an export policy must be configured on the device where the prefix export has to take place. The export policy evaluates each route to verify that it satisfies all the match conditions under the `from` statement. It also searches for the existence of the route defined under the `condition` statement (also configured under the `from` statement).

If the route does not match the entire set of required conditions defined in the policy, or if the route defined under the `condition` statement does not exist in the routing table, the route is not exported to its BGP peers. Thus, a conditional export policy matches the routes for the desired route or prefix you want installed in the peers' routing table.

To configure the conditional installation of prefixes with the help of an export policy:

1. Create a `condition` statement to check prefixes.

```
[edit]
policy-options {
    condition condition-name {
        if-route-exists address table table-name;
    }
}
```

2. Create an export policy with the newly created condition using the `condition` statement.

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

```
        term 1 {
            from {
                protocols bgp;
                condition condition-name;
            }
            then {
                accept;
            }
        }
    }
}
```

3. Apply the export policy to the device that requires only selected prefixes to be exported from the routing table.

```
[edit]
protocols bgp {
    group group-name {
        export policy-name;
    }
}
```

SEE ALSO

Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases | 480

Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table

IN THIS SECTION

- Requirements | 485
- Overview | 485
- Configuration | 489

- Verification | 499

This example shows how to configure conditional installation of prefixes in a routing table using BGP export policy.

Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers
- Junos OS Release 9.0 or later

Overview

IN THIS SECTION

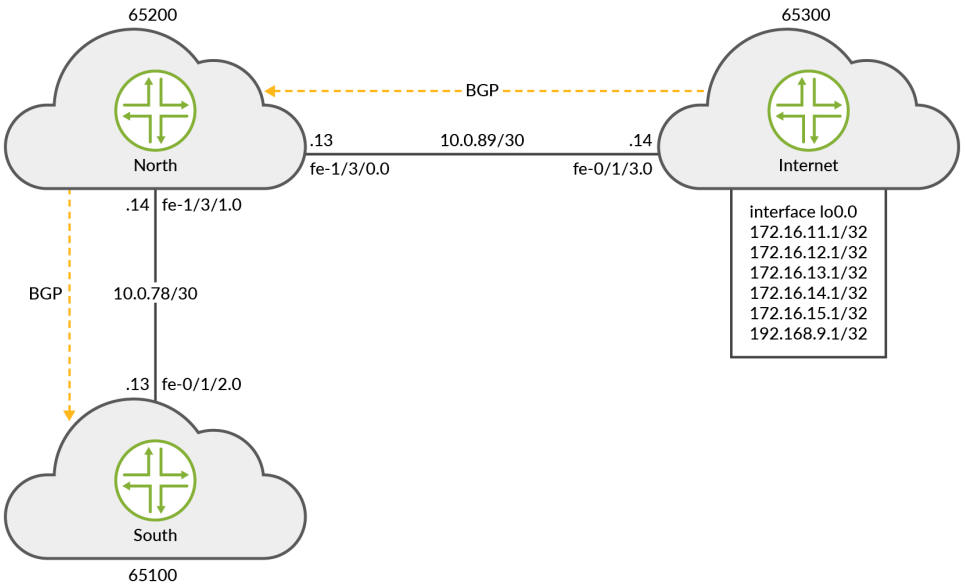
- Topology | 488

In this example, three routers in three different autonomous systems (ASs) are connected and configured with the BGP protocol. The router labeled Internet, which is the upstream router, has five addresses configured on its lo0.0 loopback interface (172.16.11.1/32, 172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, and 172.16.15.1/32), and an extra loopback address (192.168.9.1/32) is configured as the router ID. These six addresses are exported into BGP to emulate the contents of a BGP routing table of a router connected to the Internet, and advertised to North.

The North and South routers use the 10.0.89.12/30 and 10.0.78.12/30 networks, respectively, and use 192.168.7.1 and 192.168.8.1 for their respective loopback addresses.

Figure 38 on page 486 shows the topology used in this example.

Figure 38: Conditional Installation of Prefixes



Router North exports the 172.16.0.0/16 BGP routes it learns from Router Internet to Router South. These routes might represent the routes owned by the Internet router's domain. In addition, when the specific 172.16.11.1/32 route is present, Router North also advertises a default route. The 172.16.11.1 route might represent the Internet router's link to a tier 1 transit peering provider that provides full internet connectivity.

Router South receives all six routes, but should only install the default route and one other specific route (172.16.11.1/32) in its routing table.

To summarize, the example meets the following requirements:

- On North, send all 172.16/16 prefixes. In addition, also send 0/0 to South only if a particular route is present in the inet.0 routing table (in this example 172.16.11.1/32).
- On South, accept and install the default route and the 172.16.11.1/32 route in the routing and forwarding tables. Drop all other routes. Consider that South might be a lower-end device that cannot accept a full Internet routing table. As a result the operator only wants South to have the default route and one other specific prefix.

The first requirement is met with an export policy on North:

```
user@North# show policy-options
policy-statement conditional-export-bgp {
    term prefix_11 {
```

```
        from {
            protocol bgp;
            route-filter 172.16.0.0/16 orlonger;
        }
        then accept;
    }
    term conditional-default {
        from {
            route-filter 0.0.0.0/0 exact;
            condition prefix_11;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
condition prefix_11 {
    if-route-exists {
        172.16.11.1/32;
        table inet.0;
    }
}
```

The logic of the conditional export policy can be summarized as follows: If 0/0 is present, and if 172.16.11.1/32 is present, then send the 0/0 prefix. This implies that if 172.16.11.1/32 is not present, then do not send 0/0.

The second requirement is met with an import policy on South:

```
user@South# show policy-options
policy-statement import-selected-routes {
    term 1 {
        from {
            rib inet.0;
            neighbor 10.0.78.14;
            route-filter 0.0.0.0/0 exact;
            route-filter 172.16.11.1/32 exact;
        }
        then accept;
    }
    term 2 {
```

```
        then reject;
    }
}
```

In this example, four routes are dropped as a result of the import policy on South. This is because the export policy on North leaks all of the routes received from Internet, and the import policy on South excludes some of these routes.

It is important to understand that in Junos OS, although an import policy (inbound route filter) might reject a route, not use it for traffic forwarding, and not include it in an advertisement to other peers, the router retains these routes as hidden routes. These hidden routes are not available for policy or routing purposes. However, they do occupy memory space on the router. A service provider filtering routes to control the amount of information being kept in memory and processed by a router might want the router to entirely drop the routes being rejected by the import policy.

Hidden routes can be viewed by using the `show route receive-protocol bgp` *`neighbor-address`* `hidden` command. The hidden routes can then be retained or dropped from the routing table by configuring the `keep all | none` statement at the `[edit protocols bgp]` or `[edit protocols bgp group` *`group-name`*`]` hierarchy level.

The rules of BGP route retention are as follows:

- By default, all routes learned from BGP are retained, except those where the AS path is looped. (The AS path includes the local AS.)
- By configuring the `keep all` statement, all routes learned from BGP are retained, even those with the local AS in the AS path.
- By configuring the `keep none` statement, BGP discards routes that were received from a peer and that were rejected by import policy or other sanity checking. When this statement is configured and the inbound policy changes, Junos OS re-advertises all the routes advertised by the peer.

When you configure `keep all` or `keep none` and the peers support route refresh, the local speaker sends a refresh message and performs an import evaluation. For these peers, the sessions do not restart. To determine if a peer supports refresh, check for `Peer supports Refresh capability` in the output of the `show bgp neighbor` command.

CAUTION: If you configure `keep all` or `keep none` and the peer does not support session restart, the associated BGP sessions are restarted (flapped).

Topology

Configuration

IN THIS SECTION

- CLI Quick Configuration | 489
- Configuring Conditional Installation of Prefixes | 491
- Results | 494

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 Internet

```
set interfaces lo0 unit 0 family inet address 172.16.11.1/32
set interfaces lo0 unit 0 family inet address 172.16.12.1/32
set interfaces lo0 unit 0 family inet address 172.16.13.1/32
set interfaces lo0 unit 0 family inet address 172.16.14.1/32
set interfaces lo0 unit 0 family inet address 172.16.15.1/32
set interfaces lo0 unit 0 family inet address 192.168.9.1/32
set interfaces fe-0/1/3 unit 0 family inet address 10.0.89.14/30
set protocols bgp group toNorth local-address 10.0.89.14
set protocols bgp group toNorth peer-as 65200
set protocols bgp group toNorth neighbor 10.0.89.13
set protocols bgp group toNorth export into-bgp
set policy-options policy-statement into-bgp term 1 from interface lo0.0
set policy-options policy-statement into-bgp term 1 then accept
set routing-options router-id 192.168.9.1
set routing-options autonomous-system 65300
```

Router North

```
set interfaces fe-1/3/1 unit 0 family inet address 10.0.78.14/30
set interfaces fe-1/3/0 unit 0 family inet address 10.0.89.13/30
set interfaces lo0 unit 0 family inet address 192.168.8.1/32
set protocols bgp group toInternet local-address 10.0.89.13
```

```
set protocols bgp group toInternet peer-as 65300
set protocols bgp group toInternet neighbor 10.0.89.14
set protocols bgp group toSouth local-address 10.0.78.14
set protocols bgp group toSouth export conditional-export-bgp
set protocols bgp group toSouth peer-as 65100
set protocols bgp group toSouth neighbor 10.0.78.13
set policy-options policy-statement conditional-export-bgp term prefix_11 from protocol bgp
set policy-options policy-statement conditional-export-bgp term prefix_11 from route-filter
172.16.0.0/16 orlonger
set policy-options policy-statement conditional-export-bgp term prefix_11 then accept
set policy-options policy-statement conditional-export-bgp term conditional-default from route-
filter 0.0.0.0/0 exact
set policy-options policy-statement conditional-export-bgp term conditional-default from
condition prefix_11
set policy-options policy-statement conditional-export-bgp term conditional-default then accept
set policy-options policy-statement conditional-export-bgp term others then reject
set policy-options condition prefix_11 if-route-exists 172.16.11.1/32
set policy-options condition prefix_11 if-route-exists table inet.0
set routing-options static route 0/0 reject
set routing-options router-id 192.168.8.1
set routing-options autonomous-system 65200
```

Router South

```
set interfaces fe-0/1/2 unit 0 family inet address 10.0.78.13/30
set interfaces lo0 unit 0 family inet address 192.168.7.1/32
set protocols bgp group toNorth local-address 10.0.78.13
set protocols bgp group toNorth import import-selected-routes
set protocols bgp group toNorth peer-as 65200
set protocols bgp group toNorth neighbor 10.0.78.14
set policy-options policy-statement import-selected-routes term 1 from neighbor 10.0.78.14
set policy-options policy-statement import-selected-routes term 1 from route-filter
172.16.11.1/32 exact
set policy-options policy-statement import-selected-routes term 1 from route-filter 0.0.0.0/0
exact
set policy-options policy-statement import-selected-routes term 1 then accept
set policy-options policy-statement import-selected-routes term 2 then reject
set routing-options router-id 192.168.7.1
set routing-options autonomous-system 65100
```

Configuring Conditional Installation of Prefixes

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 Junos OS CLI User Guide.

To configure conditional installation of prefixes:

1. Configure the router interfaces forming the links between the three routers.

```
Router Internet
[edit interfaces]
user@Internet# set fe-0/1/3 unit 0 family inet address 10.0.89.14/30
```

```
Router North
[edit interfaces]
user@North# set fe-1/3/1 unit 0 family inet address 10.0.78.14/30
user@North# set fe-1/3/0 unit 0 family inet address 10.0.89.13/30
```

```
Router South
[edit interfaces]
user@South# set fe-0/1/2 unit 0 family inet address 10.0.78.13/30
```

2. Configure five loopback interface addresses on Router Internet to emulate BGP routes learned from the Internet that are to be imported into the routing table of Router South, and configure an additional address (192.168.9.1/32) that will be configured as the router ID.

```
Router Internet
[edit interfaces lo0 unit 0 family inet]
user@Internet# set address 172.16.11.1/32
user@Internet# set address 172.16.12.1/32
user@Internet# set address 172.16.13.1/32
user@Internet# set address 172.16.14.1/32
user@Internet# set address 172.16.15.1/32
user@Internet# set address 192.168.9.1/32
```

Also, configure the loopback interface addresses on Routers North and South.

```
Router North
[edit interfaces lo0 unit 0 family inet]
user@North# set address 192.168.8.1/32
```

```
Router South
[edit interfaces lo0 unit 0 family inet]
user@South# set address 192.168.7.1/32
```

3. Configure the static default route on Router North to be advertised to Router South.

```
[edit routing-options]
user@North# set static route 0/0 reject
```

4. Define the condition for exporting prefixes from the routing table on Router North.

```
[edit policy-options condition prefix_11]
user@North# set if-route-exists 172.16.11.1/32
user@North# set if-route-exists table inet.0
```

5. Define export policies (`into-bgp` and `conditional-export-bgp`) on Routers Internet and North respectively, to advertise routes to BGP.

NOTE: Ensure that you reference the condition, `prefix_11` (configured in Step "4" on page 492), in the export policy.

```
Router Internet
[edit policy-options policy-statement into-bgp ]
user@Internet# set term 1 from interface lo0.0
user@Internet# set term 1 then accept
```

```
Router North
[edit policy-options policy-statement conditional-export-bgp]
user@North# set  term prefix_11 from protocol bgp
```

```
user@North# set  term prefix_11 from route-filter 172,16.0.0/16 orlonger
user@North# set  term prefix_11 then accept
user@North# set  term conditional-default from route-filter 0.0.0.0/0 exact
user@North# set  term conditional-default from condition prefix_11
user@North# set  term conditional-default then accept
user@North# set term others then reject
```

6. Define an import policy (`import-selected-routes`) on Router South to import some of the routes advertised by Router North into its routing table.

```
[edit policy-options policy-statement import-selected-routes ]
user@South# set term 1 from neighbor 10.0.78.14
user@South# set term 1 from route-filter 172.16.11.1/32 exact
user@South# set term 1 from route-filter 0.0.0.0/0 exact
user@South# set term 1 then accept
user@South# set term 2 then reject
```

7. Configure BGP on all three routers to enable the flow of prefixes between the autonomous systems.

NOTE: Ensure that you apply the defined import and export policies to the respective BGP groups for prefix advertisement to take place.

```
Router Internet
[edit protocols bgp group toNorth]
user@Internet# set local-address 10.0.89.14
user@Internet# set peer-as 65200
user@Internet# set neighbor 10.0.89.13
user@Internet# set export into-bgp
```

```
Router North
[edit protocols bgp group toInternet]
user@North# set local-address 10.0.89.13
user@North# set peer-as 65300
user@North# set neighbor 10.0.89.14
```

```
[edit protocols bgp group toSouth]
user@North# set local-address 10.0.78.14
```

```
user@North# set peer-as 65100
user@North# set neighbor 10.0.78.13
user@North# set export conditional-export-bgp
```

```
Router South
[edit protocols bgp group toNorth]
user@South# set local-address 10.0.78.13
user@South# set peer-as 65200
user@South# set neighbor 10.0.78.14
user@South# set import import-selected-routes
```

8. Configure the router ID and autonomous system number for all three routers.

NOTE: In this example, the router ID is configured based on the IP address configured on the lo0.0 interface of the router.

```
Router Internet
[edit routing options]
user@Internet# set router-id 192.168.9.1
user@Internet# set autonomous-system 65300
```

```
Router North
[edit routing options]
user@North# set router-id 192.168.8.1
user@North# set autonomous-system 65200
```

```
Router South
[edit routing options]
user@South# set router-id 192.168.7.1
user@South# set autonomous-system 65100
```

Results

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

Device Internet

```
user@Internet# show interfaces
fe-0/1/3 {
    unit 0 {
        family inet {
            address 10.0.89.14/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.11.1/32;
            address 172.16.12.1/32;
            address 172.16.13.1/32;
            address 172.16.14.1/32;
            address 172.16.15.1/32;
            address 192.168.9.1/32;
        }
    }
}
```

```
user@Internet# show protocols bgp
group toNorth {
    local-address 10.0.89.14;
    export into-bgp;
    peer-as 65200;
    neighbor 10.0.89.13;
}
```

```
user@Internet# show policy-options
policy-statement into-bgp {
    term 1 {
        from interface lo0.0;
        then accept;
```

```
    }
}
```

```
user@Internet# show routing-options
router-id 192.168.9.1;
autonomous-system 65300;
```

Device North

```
user@North# show interfaces
fe-1/3/1 {
    unit 0 {
        family inet {
            address 10.0.78.14/30;
        }
    }
}
fe-1/3/0 {
    unit 0 {
        family inet {
            address 10.0.89.13/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.8.1/32;
        }
    }
}
```

```
user@North# show protocols bgp
group toInternet {
    local-address 10.0.89.13;
    peer-as 65300;
    neighbor 10.0.89.14;
}
group toSouth {
```

```
    local-address 10.0.78.14;
    export conditional-export-bgp;
    peer-as 65100;
    neighbor 10.0.78.13;
}
```

```
user@North# show policy-options
policy-statement conditional-export-bgp {
    term prefix_11 {
        from {
            protocol bgp;
            route-filter 172.16.0.0/16 orlonger;
        }
        then accept;
    }
    term conditional-default {
        from {
            route-filter 0.0.0.0/0 exact;
            condition prefix_11;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
condition prefix_11 {
    if-route-exists {
        172.16.11.1/32;
        table inet.0;
    }
}
```

```
user@North# show routing-options
static {
    route 0.0.0.0/0 reject;
}
router-id 192.168.8.1;
autonomous-system 65200;
```

Device South

```
user@South# show interfaces
fe-0/1/2 {
    unit 0 {
        family inet {
            address 10.0.78.13/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.7.1/32;
        }
    }
}
```

```
user@South# show protocols bgp
bgp {
    group toNorth {
        local-address 10.0.78.13;
        import import-selected-routes;
        peer-as 65200;
        neighbor 10.0.78.14;
    }
}
```

```
user@South# show policy-options
policy-statement import-selected-routes {
    term 1 {
        from {
            neighbor 10.0.78.14;
            route-filter 172.16.11.1 exact;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 2 {
```

```
        then reject;
    }
}
```

```
user@South# show routing-options
router-id 192.168.7.1;
autonomous-system 65100;
```

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

Verification

IN THIS SECTION

- Verifying BGP | **499**
- Verifying Prefix Advertisement from Router Internet to Router North | **502**
- Verifying Prefix Advertisement from Router North to Router South | **503**
- Verifying BGP Import Policy for Installation of Prefixes | **504**
- Verifying Conditional Export from Router North to Router South | **505**
- Verifying the Presence of Routes Hidden by Policy (Optional) | **506**

Confirm that the configuration is working properly.

Verifying BGP

Purpose

Verify that BGP sessions have been established between the three routers.

Action

From operational mode, run the `show bgp neighbor` *`neighbor-address`* command.

1. Check the BGP session on Router Internet to verify that Router North is a neighbor.

```
user@Internet> show bgp neighbor 10.0.89.13
Peer: 10.0.89.13+179 AS 65200    Local: 10.0.89.14+56187 AS 65300
  Type: External    State: Established    Flags: [ImportEval Sync]
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ into-bgp ]
  Options: [Preference LocalAddress PeerAS Refresh]
  Local Address: 10.0.89.14 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.8.1     Local ID: 192.168.9.1       Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  Local Interface: fe-0/1/3.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 65200)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 9    Sent 18   Checked 28
  Input messages:  Total 12     Updates 1       Refreshes 0     Octets 232
  Output messages: Total 14     Updates 1       Refreshes 0     Octets 383
  Output Queue[0]: 0
```

2. Check the BGP session on Router North to verify that Router Internet is a neighbor.

```
user@North> show bgp neighbor 10.0.89.14
Peer: 10.0.89.14+56187 AS 65300  Local: 10.0.89.13+179 AS 65200
  Type: External    State: Established    Flags: [ImportEval Sync]
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: [Preference LocalAddress PeerAS Refresh]
  Local Address: 10.0.89.13 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.9.1     Local ID: 192.168.8.1       Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/3/0.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 65300)
  Peer does not support Addpath
  Table inet.0 Bit: 10001
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              6
    Received prefixes:            6
    Accepted prefixes:            6
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 14   Sent 3    Checked 3
  Input messages:  Total 16     Updates 2       Refreshes 0     Octets 402
  Output messages: Total 15     Updates 0       Refreshes 0     Octets 348
  Output Queue[0]: 0
```

Check the following fields in these outputs to verify that BGP sessions have been established:

- **Peer**—Check if the peer AS number is listed.

- **Local**—Check if the local AS number is listed.
- **State**—Ensure that the value is **Established**. If not, check the configuration again and see `show bgp neighbor` for more details on the output fields.

Similarly, verify that Routers North and South form peer relationships with each other.

Meaning

BGP sessions are established between the three routers.

Verifying Prefix Advertisement from Router Internet to Router North

Purpose

Verify that the routes sent from Router Internet are received by Router North.

Action

1. From operational mode on Router Internet, run the `show route advertising-protocol bgp` *neighbor-address* command.

```
user@Internet> show route advertising-protocol bgp 10.0.89.13
inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.11.1/32            Self                                    I
* 172.16.12.1/32            Self                                    I
* 172.16.13.1/32            Self                                    I
* 172.16.14.1/32            Self                                    I
* 172.16.15.1/32            Self                                    I
* 192.168.9.1/32            Self                                    I
```

   The output verifies that Router Internet advertises the routes 172.16.11.1/32, 172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, 172.16.15.1/32, and 192.168.9.1/32 (the loopback address used as router ID) to Router North.

2. From operational mode on Router North, run the `show route receive-protocol bgp` *neighbor-address* command.

```
user@North> show route receive-protocol bgp 10.0.89.14
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
```

```
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.11.1/32            10.0.89.14                            65300 I
* 172.16.12.1/32            10.0.89.14                            65300 I
* 172.16.13.1/32            10.0.89.14                            65300 I
* 172.16.14.1/32            10.0.89.14                            65300 I
* 172.16.15.1/32            10.0.89.14                            65300 I
* 192.168.9.1/32            10.0.89.14                            65300 I
```

The output verifies that Router North has received all the routes advertised by Router Internet.

Meaning

Prefixes sent by Router Internet have been successfully installed into the routing table on Router North.

Verifying Prefix Advertisement from Router North to Router South

Purpose

Verify that the routes received from Router Internet and the static default route are advertised by Router North to Router South.

Action

1. From operational mode on Router North, run the `show route 0/0 exact` command.

```
user@North> show route 0/0 exact
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:10:22
                       Reject
```

The output verifies the presence of the static default route (0.0.0.0/0) in the routing table on Router North.

2. From operational mode on Router North, run the `show route advertising-protocol bgp` *neighbor-address* command.

```
user@North> show route advertising-protocol bgp 10.0.78.13
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
```

```
* 0.0.0.0/0              Self                                I
* 172.16.11.1/32            Self                           65300 I
* 172.16.12.1/32            Self                           65300 I
* 172.16.13.1/32            Self                           65300 I
* 172.16.14.1/32            Self                           65300 I
* 172.16.15.1/32            Self                           65300 I
```

The output verifies that Router North is advertising the static route and the 172.16.11.1/32 route received from Router Internet, as well as many other routes, to Router South.

Verifying BGP Import Policy for Installation of Prefixes

Purpose

Verify that the BGP import policy successfully installs the required prefixes.

Action

See if the import policy on Router South is operational by checking if only the static default route from Router North and the 172.16.11.1/32 route from Router South are installed in the routing table.

From operational mode, run the `show route receive-protocol bgp` *neighbor-address* command.

```
user@South> show route receive-protocol bgp 10.0.78.14
inet.0: 10 destinations, 11 routes (6 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               10.0.78.14                              65200 I
* 172.16.11.1/32          10.0.78.14                              65200 65300 I
```

The output verifies that the BGP import policy is operational on Router South, and only the static default route of 0.0.0.0/0 from Router North and the 172.16.11.1/32 route from Router Internet have leaked into the routing table on Router South.

Meaning

The installation of prefixes is successful because of the configured BGP import policy.

Verifying Conditional Export from Router North to Router South

Purpose

Verify that when Device Internet stops sending the 172.16.11.1/32 route, Device North stops sending the default 0/0 route.

Action

1. Cause Device Internet to stop sending the 172.16.11.1/32 route by deactivating the 172.16.11.1/32 address on the loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@Internet# deactivate address 172.16.11.1/32
user@Internet# commit
```

2. From operational mode on Router North, run the `show route advertising-protocol bgp` *neighbor-address* command.

```
user@North> show route advertising-protocol bgp 10.0.78.13
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.12.1/32            Self                                   65300 I
* 172.16.13.1/32            Self                                   65300 I
* 172.16.14.1/32            Self                                   65300 I
* 172.16.15.1/32            Self                                   65300 I
```

   The output verifies that Router North is not advertising the default route to Router South. This is the expected behavior when the 172.16.11.1/32 route is not present.

3. Reactivate the 172.16.11.1/32 address on Device Internet's loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@Internet# activate address 172.16.11.1/32
user@Internet# commit
```

Verifying the Presence of Routes Hidden by Policy (Optional)

Purpose

Verify the presence of routes hidden by the import policy configured on Router South.

NOTE: This section demonstrates the effects of various changes you can make to the configuration depending on your needs.

Action

View routes hidden from the routing table of Router South by:

- Using the `hidden` option for the `show route receive-protocol bgp` *`neighbor-address`* command.
- Deactivating the import policy.

1. From operational mode, run the `show route receive-protocol bgp` *`neighbor-address`* `hidden` command to view hidden routes.

```
user@South> show route receive-protocol bgp 10.0.78.14 hidden
inet.0: 10 destinations, 11 routes (6 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.12.1/32             10.0.78.14                           65200 65300 I
  172.16.13.1/32             10.0.78.14                           65200 65300 I
  172.16.14.1/32             10.0.78.14                           65200 65300 I
  172.16.15.1/32             10.0.78.14                           65200 65300 I
```

   The output verifies the presence of routes hidden by the import policy (172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, and 172.16.15.1/32) on Router South.

2. Deactivate the BGP import policy by configuring the `deactivate import` statement at the `[edit protocols bgp group` *`group-name`*`]` hierarchy level.

```
[edit protocols bgp group toNorth]
user@South# deactivate import
user@South# commit
```

3. Run the `show route receive-protocol bgp` *neighbor-address* operational mode command to check the routes after deactivating the import policy.

```
user@South> show route receive-protocol bgp 10.0.78.14
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               10.0.78.14                              65200 I
* 172.16.11.1/32          10.0.78.14                              65200 65300 I
* 172.16.12.1/32          10.0.78.14                              65200 65300 I
* 172.16.13.1/32          10.0.78.14                              65200 65300 I
* 172.16.14.1/32          10.0.78.14                              65200 65300 I
* 172.16.15.1/32          10.0.78.14                              65200 65300 I
```

   The output verifies the presence of previously hidden routes (172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, and 172.16.15.1/32).

4. Activate the BGP import policy and remove the hidden routes from the routing table by configuring the `activate import` and `keep none` statements respectively at the `[edit protocols bgp group` *group-name*`]` hierarchy level.

```
[edit protocols bgp group toNorth]
user@South# activate import
user@South# set keep none
user@South# commit
```

5. From operational mode, run the `show route receive-protocol bgp` *neighbor-address* `hidden` command to check the routes after activating the import policy and configuring the `keep none` statement.

```
user@South> show route receive-protocol bgp 10.0.78.14 hidden

inet.0: 6 destinations, 7 routes (6 active, 0 holddown, 0 hidden)
```

   The output verifies that the hidden routes are not maintained in the routing table because of the configured `keep none` statement.

SEE ALSO

Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases | 480

Conditional Advertisement and Import Policy (Routing Table) with certain match conditions | 482

Implicit filter for Default EBGP Route Propagation Behavior without Policies

SUMMARY

This section talks about using an implicit filter to regulate the EBGP route propagation behavior when there is no explicit policy configured.

IN THIS SECTION

- Benefits | 508
- Overview | 508

Benefits

This feature provides the following benefits:

- **Regulates BGP implementation**—Prevents EBGP speakers from becoming a silent pass-through where it accepted and advertised all routes by default. This feature effectively brings down the increase in transit traffic on leaf autonomous systems, especially when they are multi-homed to any upstream Internet Service Providers. Thus, it also prevents silent dropping of traffic, Denial of Service, and global internet outages.
- **Implicit filter**—The configuration facilitates the use of an implicit filter, where the default behavior is still set to receive and advertise all routes by default. The configuration statement only adds an option to specify enable or disable for accept, reject, reject-always clauses, when required. The implicit filter ensures that the users with existing deployments that rely on the default BGP policy do not experience operational disruptions.

Overview

BGP is the current inter-domain Autonomous protocol used for global Internet routing. It also supports various services such as VPNs, and link state, which are not intended for global usage.

BGP implementation, including the default EBGP behavior is guided by *RFC4271, A Border Gateway Protocol 4 (BGP-4)*. However, it does not provide any explicit guidance on specifying what routes should be distributed. This leads to the original BGP implementation being a silent pass-through for routes without any filtering and therefore, causing an increase in traffic, resulting in global Internet outages.

Starting in Junos OS Release 20.3R1, we have introduced an implicit filter `defaults ebgp no-policy` at the existing `[edit protocols bgp]` hierarchy level. The configuration separates the default policy for receive and advertise, into separate clauses (accept, reject, or reject-always) to permit the behavior to vary independently.

If there is no explicit policy configured, the implicit filter allows you to enable the default eBGP receive and advertise behavior in one of three states as follows:

Values	Default Policy	What it does
accept	receive	Accepts to receive all routes (also the default behavior).
	advertise	Accepts to advertise all routes (also the default behavior).
reject	receive	Rejects to receive routes of type inet unicast and inet6 unicast in instance types primary, vrf, virtual-router, and non-forwarding.
	advertise	Rejects to advertise routes of type inet unicast and inet6 unicast in instance types primary, vrf, virtual-router, and non-forwarding.
reject-always	receive	Rejects to receive all routes.
	advertise	Rejects to advertise all routes.

SEE ALSO

defaults

Routing Policies for BGP Communities

IN THIS SECTION

- Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions | **510**
- Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS | **512**
- Example: Configuring a Routing Policy That Removes BGP Communities | **525**

- Example: Configuring a Routing Policy Based on the Number of BGP Communities | 538

Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions

A *BGP community* is a group of destinations that share a common property. Community information is included as a path attribute in BGP update messages. This information identifies community members and enables you to perform actions on a group without having to elaborate upon each member. You can use community and extended communities attributes to trigger routing decisions, such as acceptance, rejection, preference, or redistribution.

You can assign community tags to non-BGP routes through configuration (for static, aggregate, or generated routes) or an import routing policy. These tags can then be matched when BGP exports the routes.

A community value is a 32-bit field that is divided into two main sections. The first 16 bits of the value encode the AS number of the network that originated the community, while the last 16 bits carry a unique number assigned by the AS. This system attempts to guarantee a globally unique set of community values for each AS in the Internet. Junos OS uses a notation of *as-number*:*community-value*, where each value is a decimal number. The AS values of 0 and 65,535 are reserved, as are all of the community values within those AS numbers. Each community, or set of communities, is given a name within the `[edit policy-options]` configuration hierarchy. The name of the community uniquely identifies it to the routing device and serves as the method by which routes are categorized. For example, a route with a community value of 64510:1111 might belong to the community named `AS64510-routes`. The community name is also used within a routing policy as a match criterion or as an action. The command syntax for creating a community is: policy-options `community` *`name`* `members [`*`community-ids`*`]`. The *`community-ids`* are either a single community value or multiple community values. When more than one value is assigned to a community name, the routing device interprets this as a logical AND of the community values. In other words, a route must have all of the configured values before being assigned the community name.

The regular community attribute is four octets. Networking enhancements, such as VPNs, have functionality requirements that can be satisfied by an attribute such as a community. However, the 4-octet community value does not provide enough expansion and flexibility to accommodate VPN requirements. This leads to the creation of extended communities. An extended community is an 8-octet value that is also divided into two main sections. The first 2 octets of the community encode a type field while the last 6 octets carry a unique set of data in a format defined by the type field. Extended communities provide a larger range for grouping or categorizing communities.

The BGP extended communities attribute format has three fields: *type*:*administrator*:*assigned-number*. The routing device expects you to use the words `target` or `origin` to represent the type field. The administrator field uses a decimal number for the AS or an IPv4 address, while the assigned number field expects a decimal number no larger than the size of the field (65,535 for 2 octets or 4,294,967,295 for 4 octets).

When specifying community IDs for standard and extended community attributes, you can use UNIX-style regular expressions. The only exception is for VPN import policies (`vrf-import`), which do not support regular expressions for the extended communities attribute.

Regular BGP communities attributes are a variable length attribute consisting of a set of one or more 4-byte values that was split into 16 bit values. The most significant word is interpreted as an AS number and least significant word is a locally defined value assigned by the operator of the AS. Since the adoption of 4-byte ASNs, the 4-byte BGP regular community and 6-byte BGP extended community can no longer support BGP community attributes. Operators often encode AS number in the local portion of the BGP community that means that sometimes the format of the community is ASN:ASN. With the 4-byte ASN , you need 8-bytes to encode it. Although BGP extended community permits a 4-byte AS to be encoded as the global administrator field, the local administrator field has only 2-byte of available space. Thus, 6-byte extended community attribute is also unsuitable. To overcome this, Junos OS allows you to configure optional transitive path attribute - a 12-byte BGP large community that provides the most significant 4-byte value to encode autonomous system number as the global administrator and the remaining two 4-byte assigned numbers to encode the local values as defined in RFC 8092. You can configure BGP large community at the `[edit policy-options community` *`community-name`* `members]` and `[edit routing-options static route` *`ip-address`* `community]` hierarchy levels. The BGP large community attributes format has four fields: `large`:*`global administrator`*:*`assigned number`*:*`assigned number`*.

The BGP IPv6 unicast address specific extended community are encoded as a set of 20-bytes value. The 20-byte value gets interpreted in the following format:

- Most significant 2-bytes encodes the Type and Sub-Type value (high value (most significant byte) and Low value (second most significant byte)).
- Next 16-bytes encodes the IPv6 unicast address. It is the global administrator in the IETF RFC.
- Last 2-bytes encodes the operator defined local values. It is local administrator in the IETF RFC.

The IPv6 unicast address specific BGP extended community attributes are represented by a keyword `ipv6-target`, `ipv6-origin`, or `ipv6-extended` followed by IPv6 and local administrator separated by <, >, and :.

NOTE: The length of the BGP large communities attribute value should be a non-zero multiple of 12.

SEE ALSO

Understanding How to Define BGP Communities and Extended Communities

How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions

Example: Configuring a Routing Policy That Removes BGP Communities | **525**

Example: Configuring Communities in a Routing Policy

Example: Configuring Extended Communities in a Routing Policy

Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS

IN THIS SECTION

- Requirements | **512**
- Overview | **512**
- Configuration | **513**
- Verification | **524**

This example defines a policy that takes BGP routes from the `Edu` community and places them into IS-IS with a metric of 63.

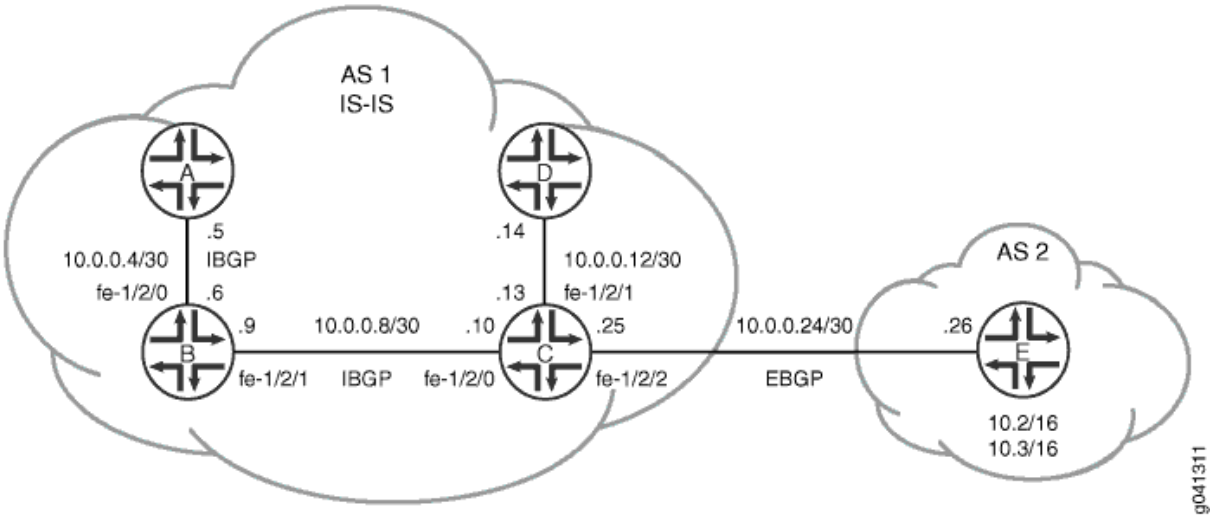
Requirements

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

Overview

Figure 39 on page 513 shows the topology used in this example.

Figure 39: Redistributing BGP Routes with a Specific Community Tag into IS-IS



In this example, Device A, Device B, Device C, and Device D are in autonomous system (AS) 1 and are running IS-IS. All of the AS 1 devices, except Device D, are running internal BGP (IBGP).

Device E is in AS 2 and has an external BGP (EBGP) peering session with Device C. Device E has two static routes, 10.2.0.0/16 and 10.3.0.0/16. These routes are tagged with the Edu 2:5 community attribute and are advertised by way of EBGP to Device C.

Device C accepts the BGP routes that are tagged with the Edu 2:5 community attribute, redistributes the routes into IS-IS, and applies an IS-IS metric of 63 to these routes.

"CLI Quick Configuration" on page 514 shows the configuration for all of the devices in Figure 39 on page 513. The section "No Link Title" on page 517 describes the steps on Device C and Device E.

Configuration

IN THIS SECTION

- Procedure | 514

Procedure

CLI Quick Configuration

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

Device A

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0001.00
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols isis interface fe-1/2/0.0 level 1 disable
set protocols isis interface lo0.0
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device B

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.6/30
set interfaces fe-1/2/0 unit 0 family iso
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces fe-1/2/1 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0002.00
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.3
set protocols isis interface fe-1/2/0.0 level 1 disable
set protocols isis interface fe-1/2/1.0 level 1 disable
set protocols isis interface lo0.0
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 1
```

Device C

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.10/30
set interfaces fe-1/2/0 unit 0 family iso
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.13/30
set interfaces fe-1/2/1 unit 0 family iso
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.25/30
set interfaces fe-1/2/2 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0003.00
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group external-peers type external
set protocols bgp group external-peers export send-isis-and-direct
set protocols bgp group external-peers peer-as 2
set protocols bgp group external-peers neighbor 10.0.0.26
set protocols isis export Edu-to-isis
set protocols isis interface fe-1/2/0.0 level 1 disable
set protocols isis interface fe-1/2/1.0 level 1 disable
set protocols isis interface fe-1/2/2.0 level 1 disable
set protocols isis interface fe-1/2/2.0 level 2 passive
set protocols isis interface lo0.0
set policy-options policy-statement Edu-to-isis term 1 from protocol bgp
set policy-options policy-statement Edu-to-isis term 1 from community Edu
set policy-options policy-statement Edu-to-isis term 1 then metric 63
set policy-options policy-statement Edu-to-isis term 1 then accept
set policy-options policy-statement send-isis-and-direct term 1 from protocol isis
set policy-options policy-statement send-isis-and-direct term 1 from protocol direct
set policy-options policy-statement send-isis-and-direct term 1 from route-filter 10.0.0.0/16
orlonger
set policy-options policy-statement send-isis-and-direct term 1 from route-filter 192.168.0.0/16
orlonger
set policy-options policy-statement send-isis-and-direct term 1 then accept
set policy-options community Edu members 2:5
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 1
```

Device D

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.14/30
set interfaces fe-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set interfaces lo0 unit 0 family iso address 49.0002.0192.0168.0004.00
set protocols isis interface fe-1/2/0.0 level 1 disable
set protocols isis interface lo0.0
set routing-options router-id 192.168.0.4
set routing-options autonomous-system 1
```

Device E

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.26/30
set interfaces lo0 unit 7 family inet address 192.168.0.5/32 primary
set interfaces lo0 unit 7 family inet address 10.2.0.1/32
set interfaces lo0 unit 7 family inet address 10.3.0.1/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers export statics
set protocols bgp group external-peers peer-as 1
set protocols bgp group external-peers neighbor 10.0.0.25
set policy-options policy-statement statics from protocol static
set policy-options policy-statement statics then community add Edu
set policy-options policy-statement statics then accept
set policy-options community Edu members 2:5
set routing-options static route 10.2.0.0/16 reject
set routing-options static route 10.2.0.0/16 install
set routing-options static route 10.3.0.0/16 reject
set routing-options static route 10.3.0.0/16 install
set routing-options router-id 192.168.0.5
set routing-options autonomous-system 2
```

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 E:

1. Configure the interfaces.

```
[edit interfaces]
user@E# set fe-1/2/0 unit 0 family inet address 10.0.0.26/30
user@E# set lo0 unit 7 family inet address 192.168.0.5/32 primary
user@E# set lo0 unit 7 family inet address 10.2.0.1/32
user@E# set lo0 unit 7 family inet address 10.3.0.1/32
```

2. Configure the `statics` policy, which adds the `Edu` community attribute to the static routes.

```
[edit policy-options]
user@E# set policy-statement statics from protocol static
user@E# set policy-statement statics then community add Edu
user@E# set policy-statement statics then accept
user@E# set community Edu members 2:5
```

3. Configure EBGP and apply the `statics` policy.

```
[edit protocols bgp group external-peers]
user@E# set type external
user@E# set export statics
user@E# set peer-as 1
user@E# set protocols bgp group external-peers neighbor 10.0.0.25
```

4. Configure the static routes.

```
[edit routing-options static]
user@E# set route 10.2.0.0/16 reject
user@E# set route 10.2.0.0/16 install
user@E# set route 10.3.0.0/16 reject
user@E# set route 10.3.0.0/16 install
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@E# set router-id 192.168.0.5
user@E# set autonomous-system 2
```

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 C:

1. Configure the interfaces.

```
[edit interfaces]
user@C# set fe-1/2/0 unit 0 family inet address 10.0.0.10/30
user@C# set fe-1/2/0 unit 0 family iso
user@C# set fe-1/2/1 unit 0 family inet address 10.0.0.13/30
user@C# set fe-1/2/1 unit 0 family iso
user@C# set fe-1/2/2 unit 0 family inet address 10.0.0.25/30
user@C# set fe-1/2/2 unit 0 family iso
user@C# set lo0 unit 0 family inet address 192.168.0.3/32
user@C# set lo0 unit 0 family iso address 49.0002.0192.0168.0003.00
```

2. Configure IBGP.

```
[edit protocols bgp group int]
user@C# set type internal
user@C# set local-address 192.168.0.3
user@C# set neighbor 192.168.0.1
user@C# set neighbor 192.168.0.2
```

3. Configure the Edu-to-isis policy, which redistributes the Edu-tagged BGP routes learned from Device E and applies a metric of 63.

```
[edit policy-options]
user@C# set policy-statement Edu-to-isis term 1 from protocol bgp
user@C# set policy-statement Edu-to-isis term 1 from community Edu
user@C# set policy-statement Edu-to-isis term 1 then metric 63
user@C# set policy-statement Edu-to-isis term 1 then accept
user@C# set community Edu members 2:5
```

4. Enable IS-IS on the interfaces, and apply the Edu-to-isis policy.

```
[edit protocols isis]
user@C# set export Edu-to-isis
user@C# set interface fe-1/2/0.0 level 1 disable
user@C# set interface fe-1/2/1.0 level 1 disable
user@C# set interface fe-1/2/2.0 level 1 disable
user@C# set interface fe-1/2/2.0 level 2 passive
user@C# set interface lo0.0
```

5. Configure the send-isis-and-direct policy, which redistributes routes to Device E, through EBGP.

   Without this policy, Device E would not have connectivity to the networks in AS 1.

```
[edit policy-options policy-statement send-isis-and-direct term 1]
user@C# set from protocol isis
user@C# set from protocol direct
user@C# set from route-filter 10.0.0.0/16 orlonger
user@C# set from route-filter 192.168.0.0/16 orlonger
user@C# set then accept
```

6. Configure EBGP and apply the send-isis-and-direct policy.

```
[edit protocols bgp group external-peers]
user@C# set type external
user@C# set export send-isis-and-direct
user@C# set peer-as 2
user@C# set neighbor 10.0.0.26
```

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

```
[edit routing-options]
user@C# set router-id 192.168.0.3
user@C# set autonomous-system 1
```

Results

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

Device E

```
user@E# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.26/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.5/32 {
                primary;
            }
            address 10.2.0.1/32;
            address 10.3.0.1/32;
        }
    }
}
```

```
user@E# show protocols
bgp {
    group external-peers {
        type external;
        export statics;
        peer-as 1;
        neighbor 10.0.0.25;
    }
}
```

```
user@E# show policy-options
policy-statement statics {
```

```
    from protocol static;
    then {
        community add Edu;
        accept;
    }
}
community Edu members 2:5;
```

```
user@E# show routing-options
static {
    route 10.2.0.0/16 {
        reject;
        install;
    }
    route 10.3.0.0/16 {
        reject;
        install;
    }
}
router-id 192.168.0.5;
autonomous-system 2;
```

Device C

```
user@C# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.10/30;
        }
        family iso;
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.0.0.13/30;
        }
        family iso;
    }
}
```

```
fe-1/2/2 {
    unit 0 {
        family inet {
            address 10.0.0.25/30;
        }
        family iso;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.3/32;
        }
        family iso {
            address 49.0002.0192.0168.0003.00;
        }
    }
}
```

```
user@C# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.3;
        neighbor 192.168.0.1;
        neighbor 192.168.0.2;
    }
    group external-peers {
        type external;
        export send-isis-and-direct;
        peer-as 2;
        neighbor 10.0.0.26;
    }
}
isis {
    export Edu-to-isis;
    interface fe-1/2/0.0 {
        level 1 disable;
    }
    interface fe-1/2/1.0 {
        level 1 disable;
```

```
    }
    interface fe-1/2/2.0 {
        level 1 disable;
        level 2 passive;
    }
    interface lo0.0;
}
```

```
user@C# show policy-options
policy-statement Edu-to-isis {
    term 1 {
        from {
            protocol bgp;
            community Edu;
        }
        then {
            metric 63;
            accept;
        }
    }
}
policy-statement send-isis-and-direct {
    term 1 {
        from {
            protocol [ isis direct ];
            route-filter 10.0.0.0/16 orlonger;
            route-filter 192.168.0.0/16 orlonger;
        }
        then accept;
    }
}
community Edu members 2:5;
```

```
user@C# show routing-options
router-id 192.168.0.3;
autonomous-system 1;
```

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

Verification

IN THIS SECTION

- Verifying the IS-IS Neighbor | 524

Confirm that the configuration is working properly.

Verifying the IS-IS Neighbor

Purpose

Verify that the BGP routes from Device E are communicated on the IS-IS network in AS 1.

Action

From operational mode, enter the `show route protocol isis` command.

```
user@D> show route protocol isis
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[IS-IS/18] 22:30:53, metric 30
                    > to 10.0.0.13 via fe-1/2/0.0
10.0.0.8/30        *[IS-IS/18] 22:30:53, metric 20
                    > to 10.0.0.13 via fe-1/2/0.0
10.0.0.24/30       *[IS-IS/18] 03:31:21, metric 20
                    > to 10.0.0.13 via fe-1/2/0.0
10.2.0.0/16        *[IS-IS/165] 02:36:31, metric 73
                    > to 10.0.0.13 via fe-1/2/0.0
10.3.0.0/16        *[IS-IS/165] 02:36:31, metric 73
                    > to 10.0.0.13 via fe-1/2/0.0
192.168.0.1/32     *[IS-IS/18] 03:40:28, metric 30
                    > to 10.0.0.13 via fe-1/2/0.0
192.168.0.2/32     *[IS-IS/18] 22:30:53, metric 20
                    > to 10.0.0.13 via fe-1/2/0.0
192.168.0.3/32     *[IS-IS/18] 22:30:53, metric 10
                    > to 10.0.0.13 via fe-1/2/0.0
```

```
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Meaning

As expected, the 10.2.0.0/16 and 10.3.0.0/16 routes are in Device D's routing table as IS-IS external routes with a metric of 73. If Device C had not added 63 to the metric, Device D would have a metric of 10 for these routes.

SEE ALSO

Advertising LSPs into IGPs

Example: Configuring a Routing Policy That Removes BGP Communities

IN THIS SECTION

- Requirements | 525
- Overview | 526
- Configuration | 527
- Verification | 534

This example shows how to create a policy that accepts BGP routes, but removes BGP communities from the routes.

Requirements

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

Overview

IN THIS SECTION

- Topology | 526

This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send two static routes to Device R1. On Device R1, an import policy specifies that all BGP communities must be removed from the routes.

By default, when communities are configured on EBGP peers, they are sent and accepted. To suppress the acceptance of communities received from a neighbor, you can remove all communities or a specified set of communities. When the result of a policy is an empty set of communities, the community attribute is not included. To remove all communities, first define a wildcard set of communities (here, the community is named `wild`):

```
[edit policy-options]
community wild members "* : *";
```

Then, in the routing policy statement, specify the `community delete` action:

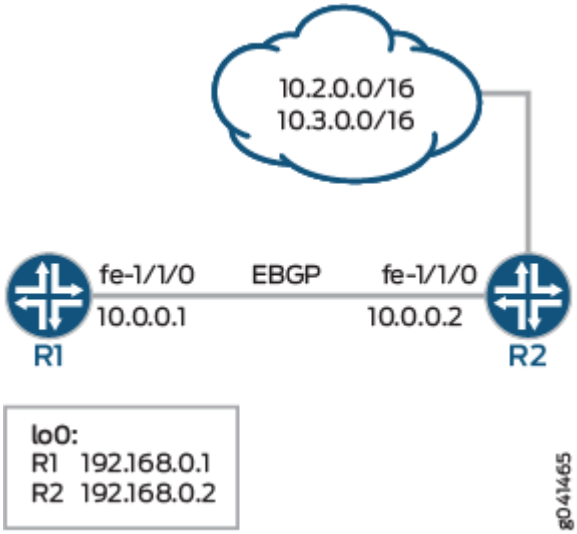
```
[edit policy-options]
policy-statement policy-name {
    term term-name {
        then community delete wild;
    }
}
```

To suppress a particular community from any autonomous system (AS), define the community as `community wild members "*:community-value"`.

Topology

Figure 40 on page 527 shows the sample network.

Figure 40: BGP Policy That Removes Communities



Configuration

IN THIS SECTION

- CLI Quick Configuration | 527
- Procedure | 529

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 R1

```
set interfaces fe-1/1/0 unit 0 description to-R2
set interfaces fe-1/1/0 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 external-peers type external
set protocols bgp group external-peers peer-as 2
set protocols bgp group external-peers neighbor 10.0.0.2 import remove-communities
set policy-options policy-statement remove-communities term 1 from protocol bgp
set policy-options policy-statement remove-communities term 1 then community delete wild
```

```
set policy-options policy-statement remove-communities term 1 then accept
set policy-options policy-statement remove-communities term 2 then reject
set policy-options community wild members *:*
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device R2

```
set interfaces fe-1/1/0 unit 0 description to-R1
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers export statics
set protocols bgp group external-peers peer-as 1
set protocols bgp group external-peers neighbor 10.0.0.1
set policy-options policy-statement statics from protocol static
set policy-options policy-statement statics then community add 1
set policy-options policy-statement statics then accept
set policy-options community 1 members 2:1
set policy-options community 1 members 2:2
set policy-options community 1 members 2:3
set policy-options community 1 members 2:4
set policy-options community 1 members 2:5
set policy-options community 1 members 2:6
set policy-options community 1 members 2:7
set policy-options community 1 members 2:8
set policy-options community 1 members 2:9
set policy-options community 1 members 2:10
set routing-options static route 10.2.0.0/16 reject
set routing-options static route 10.2.0.0/16 install
set routing-options static route 10.3.0.0/16 reject
set routing-options static route 10.3.0.0/16 install
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 2
```

Procedure

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/1/0 unit 0 description to-R2
user@R1# set fe-1/1/0 unit 0 family inet address 10.0.0.1/30
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group external-peers]
user@R1# set type external
user@R1# set peer-as 2
user@R1# set neighbor 10.0.0.2 import remove-communities
```

3. Configure the routing policy that deletes communities.

```
[edit policy-options policy-statement remove-communities]
user@R1# set term 1 from protocol bgp
user@R1# set term 1 then community delete wild
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

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

```
[edit routing-options ]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 1
```

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/1/0 unit 0 description to-R1
user@R2# set fe-1/1/0 unit 0 family inet address 10.0.0.2/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

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

```
[edit routing-options]
user@R2# set router-id 192.168.0.3
user@R2# set autonomous-system 2
```

3. Configure BGP.

```
[edit  protocols bgp group external-peers]
user@R2# set type external
user@R2# set peer-as 1
user@R2# set neighbor 10.0.0.1
```

4. Configure multiple communities, or configure a single community with multiple members.

```
[edit policy-options community 1]
user@R2# set members 2:1
user@R2# set members 2:2
user@R2# set members 2:3
user@R2# set members 2:4
user@R2# set members 2:5
user@R2# set members 2:6
user@R2# set members 2:7
user@R2# set members 2:8
```

```
user@R2# set members 2:9
user@R2# set members 2:10
```

5. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.2.0.0/16 reject
user@R2# set route 10.2.0.0/16 install
user@R2# set route 10.3.0.0/16 reject
user@R2# set route 10.3.0.0/16 install
```

6. Configure a routing policy that advertises static routes into BGP and adds the BGP community to the routes.

```
[edit policy-options policy-statement statics]
user@R2# set from protocol static
user@R2# set then community add 1
user@R2# set then accept
```

7. Apply the export policy.

```
[edit protocols bgp group external-peers]
user@R2# set export statics
```

Results

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

Device R1

```
user@R1# show interfaces
fe-1/1/0 {
    unit 0{
        description to-R2;
        family inet {
            address 10.0.0.1/30;
        }
```

```
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group external-peers {
        type external;
        peer-as 2;
        neighbor 10.0.0.2 {
            import remove-communities;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement remove-communities {
    term 1 {
        from protocol bgp;
        then {
            community delete wild;
            accept;
        }
    }
    term 2 {
        then reject;
    }
```

```
}
community wild members *:*;
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 1;
```

Device R2

```
user@R2# show interfaces
fe-1/1/0 {
    unit 0 {
        description to-R1;
        family inet {
            address 10.0.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group external-peers {
        type external;
        export statics;
        peer-as 1;
        neighbor 10.0.0.1;
    }
}
```

```
user@R2# show policy-options
policy-statement statics {
```

```
    from protocol static;
    then {
        community add 1;
        accept;
    }
}
community 1 members [ 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10 ];
```

```
user@R2# show routing-options
static {
    route 10.2.0.0/16 {
        reject;
        install;
    }
    route 10.3.0.0/16 {
        reject;
        install;
    }
}
router-id 192.168.0.3;
autonomous-system 2;
```

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

Verification

IN THIS SECTION

- Verifying the BGP Routes | 534

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose

Make sure that the routing table on Device R1 does not contain BGP communities.

Action

1. On Device R1, run the `show route protocols bgp extensive` command.

```
user@R1> show route protocols bgp extensive

inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
10.2.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:39:01
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3

10.3.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:39:01
                Validation State: unverified
```

```
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```

2. On Device R1, deactivate the `community remove` configuration in the import policy.

```
[edit policy-options policy-statement remove-communities term 1]
user@R1# deactivate then community delete wild
user@R1# commit
```

3. On Device R1, run the `show route protocols bgp extensive` command to view the advertised communities.

```
user@R1> show route protocols bgp extensive
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
10.2.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:40:53
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3

10.3.0.0/16 (1 entry, 1 announced)
```

```
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:40:53
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```

Meaning

The output shows that in Device R1's routing table, the communities are suppressed in the BGP routes sent from Device R2. When the `community remove` setting in Device R1's import policy is deactivated, the communities are no longer suppressed.

SEE ALSO

- Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS
- Understanding External BGP Peering Sessions

Example: Configuring a Routing Policy Based on the Number of BGP Communities

IN THIS SECTION

- Requirements | 538
- Overview | 538
- Configuration | 539
- Verification | 546

This example shows how to create a policy that accepts BGP routes based on the number of BGP communities.

Requirements

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

Overview

IN THIS SECTION

- Topology | 538

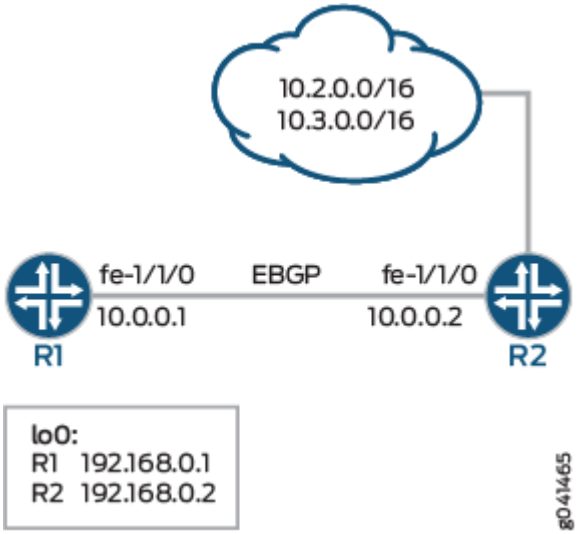
This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send two static routes to Device R1. On Device R1, an import policy specifies that the BGP-received routes can contain up to five communities to be considered a match. For example, if a route contains three communities, it is considered a match and is accepted. If a route contains six or more communities, it is considered a nonmatch and is rejected.

It is important to remember that the default policy for EBGP is to accept all routes. To ensure that the nonmatching routes are rejected, you must include a `then reject` action at the end of the policy definition.

Topology

Figure 41 on page 539 shows the sample network.

Figure 41: BGP Policy with a Limit on the Number of Communities Accepted



Configuration

IN THIS SECTION

- CLI Quick Configuration | 539
- Procedure | 540

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 R1

```
set interfaces fe-1/1/0 unit 0 description to-R2
set interfaces fe-1/1/0 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 external-peers type external
set protocols bgp group external-peers peer-as 2
set protocols bgp group external-peers neighbor 10.0.0.2 import import-communities
set policy-options policy-statement import-communities term 1 from protocol bgp
set policy-options policy-statement import-communities term 1 from community-count 5 orlower
```

```
set policy-options policy-statement import-communities term 1 then accept
set policy-options policy-statement import-communities term 2 then reject
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device R2

```
set interfaces fe-1/1/0 unit 0 description to-R1
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers export statics
set protocols bgp group external-peers peer-as 1
set protocols bgp group external-peers neighbor 10.0.0.1
set policy-options policy-statement statics from protocol static
set policy-options policy-statement statics then community add 1
set policy-options policy-statement statics then accept
set policy-options community 1 members 2:1
set policy-options community 1 members 2:2
set policy-options community 1 members 2:3
set policy-options community 1 members 2:4
set policy-options community 1 members 2:5
set policy-options community 1 members 2:6
set policy-options community 1 members 2:7
set policy-options community 1 members 2:8
set policy-options community 1 members 2:9
set policy-options community 1 members 2:10
set routing-options static route 10.2.0.0/16 reject
set routing-options static route 10.2.0.0/16 install
set routing-options static route 10.3.0.0/16 reject
set routing-options static route 10.3.0.0/16 install
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 2
```

Procedure

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/1/0 unit 0 description to-R2
user@R1# set fe-1/1/0 unit 0 family inet address 10.0.0.1/30
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group external-peers]
user@R1# set type external
user@R1# set peer-as 2
user@R1# set neighbor 10.0.0.2 import import-communities
```

3. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement import-communities]
user@R1# set term 1 from protocol bgp
user@R1# set term 1 from community-count 5 orlower
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

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

```
[edit routing-options ]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 1
```

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/1/0 unit 0 description to-R1
user@R2# set fe-1/1/0 unit 0 family inet address 10.0.0.2/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

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

```
[edit routing-options]
user@R2# set router-id 192.168.0.3
user@R2# set autonomous-system 2
```

3. Configure BGP.

```
[edit  protocols bgp group external-peers]
user@R2# set type external
user@R2# set peer-as 1
user@R2# set neighbor 10.0.0.1
```

4. Configure multiple communities, or configure a single community with multiple members.

```
[edit policy-options community 1]
user@R2# set members 2:1
user@R2# set members 2:2
user@R2# set members 2:3
user@R2# set members 2:4
user@R2# set members 2:5
user@R2# set members 2:6
user@R2# set members 2:7
user@R2# set members 2:8
user@R2# set members 2:9
user@R2# set members 2:10
```

5. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.2.0.0/16 reject
```

```
user@R2# set route 10.2.0.0/16 install
user@R2# set route 10.3.0.0/16 reject
user@R2# set route 10.3.0.0/16 install
```

6. Configure a routing policy that advertises static routes into BGP and adds the BGP community to the routes.

```
[edit policy-options policy-statement statics]
user@R2# set from protocol static
user@R2# set then community add 1
user@R2# set then accept
```

7. Apply the export policy.

```
[edit protocols bgp group external-peers]
user@R2# set export statics
```

Results

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

Device R1

```
user@R1# show interfaces
fe-1/1/0 {
    unit 0{
        description to-R2;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
```

```
}
}
```

```
user@R1# show protocols
bgp {
    group external-peers {
        type external;
        peer-as 2;
        neighbor 10.0.0.2 {
            import import-communities;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement import-communities {
    term 1 {
        from {
            protocol bgp;
            community-count 5 orlower;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 1;
```

Device R2

```
user@R2# show interfaces
fe-1/1/0 {
    unit 0 {
        description to-R1;
```

```
        family inet {
            address 10.0.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group external-peers {
        type external;
        export statics;
        peer-as 1;
        neighbor 10.0.0.1;
    }
}
```

```
user@R2# show policy-options
policy-statement statics {
    from protocol static;
    then {
        community add 1;
        accept;
    }
}
community 1 members [ 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10 ];
```

```
user@R2# show routing-options
static {
    route 10.2.0.0/16 {
        reject;
        install;
    }
```

```
    route 10.3.0.0/16 {
        reject;
        install;
    }
}
router-id 192.168.0.3;
autonomous-system 2;
```

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

Verification

IN THIS SECTION

- Verifying the BGP Routes | 546

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose

Make sure that the routing table on Device R1 contains the expected BGP routes.

Action

1. On Device R1, run the `show route protocols bgp` command.

```
user@R1> show route protocols bgp

inet.0: 5 destinations, 5 routes (3 active, 0 holddown, 2 hidden)
```

2. On Device R1, change the `community-count` configuration in the import policy.

```
[edit policy-options policy-statement import-communities term 1]
user@R1# set from community-count 5 orhigher
user@R1# commit
```

3. On Device R1, run the `show route protocols bgp` command.

```
user@R1> show route protocols bgp

inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.2.0.0/16        *[BGP/170] 18:29:53, localpref 100
                      AS path: 2 I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/1/0.0
10.3.0.0/16        *[BGP/170] 18:29:53, localpref 100
                      AS path: 2 I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/1/0.0
```

4. On Device R1, run the `show route protocols bgp extensive` command to view the advertised communities.

```
user@R1> show route protocols bgp extensive
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
10.2.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/1/0.0, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 18:56:10
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```

```
10.3.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/1/0.0, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 18:56:10
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```

Meaning

The output shows that in Device R1's routing table, the BGP routes sent from Device R2 are hidden. When the `community-count` setting in Device R1's import policy is modified, the BGP routes are no longer hidden.

SEE ALSO

- Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS
- Understanding External BGP Peering Sessions

5

CHAPTER

Enabling Load Balancing for BGP

IN THIS CHAPTER

- Load Balancing for a BGP Session | **550**
- BGP Egress Traffic Engineering | **882**
- Link-State Distribution Using BGP | **989**

Load Balancing for a BGP Session

IN THIS SECTION

- Understanding BGP Multipath | **551**
- Example: Load Balancing BGP Traffic | **552**
- Understanding Configuration of Up to 512 Equal-Cost Paths With Optional Consistent Load Balancing | **561**
- Example: Configuring Single-Hop EBGP Peers to Accept Remote Next Hops | **564**
- Understanding Load Balancing for BGP Traffic with Unequal Bandwidth Allocated to the Paths | **581**
- Example: Load Balancing BGP Traffic with Unequal Bandwidth Allocated to the Paths | **582**
- Advertising Aggregate Bandwidth Across External BGP Links for Load Balancing Overview | **594**
- Example: Configuring a Policy to Advertise Aggregate Bandwidth Across External BGP Links for Load Balancing | **596**
- Understanding the Advertisement of Multiple Paths to a Single Destination in BGP | **610**
- Example: Advertising Multiple Paths in BGP | **612**
- Example: Configuring Selective Advertising of BGP Multiple Paths for Load Balancing | **649**
- Example: Configuring a Routing Policy to Select and Advertise Multipaths Based on BGP Community Value | **667**
- Configuring Recursive Resolution over BGP Multipath | **682**
- Configuring ECMP Next Hops for RSVP and LDP LSPs for Load Balancing | **684**
- Configuring Consistent Load Balancing for ECMP Groups | **686**
- Improve Network Resiliency Using Multiple ECMP BGP Peers | **689**
- Understanding Entropy Label for BGP Labeled Unicast LSP | **691**
- Configuring an Entropy Label for a BGP Labeled Unicast LSP | **695**
- Example: Configuring an Entropy Label for a BGP Labeled Unicast LSP | **697**
- Use Case for BGP Prefix Independent Convergence for Inet, Inet6, or Labeled Unicast | **722**
- Configuring BGP Prefix Independent Convergence for Inet | **723**
- Example: Configuring BGP Prefix Independent Convergence for Inet | **728**
- BGP PIC Edge Using BGP Labeled Unicast Overview | **748**
- Configuring BGP PIC Edge Using BGP Labeled Unicast for Layer 2 Services | **750**
- Example: Protecting IPv4 Traffic over Layer 3 VPN Running BGP Labeled Unicast | **751**
- FAT Pseudowire Support for BGP L2VPN and VPLS Overview | **822**

- Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **823**
- Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **824**
- Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **859**
- Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **860**

Understanding BGP Multipath

BGP multipath allows you to install multiple internal BGP paths and multiple external BGP paths to the forwarding table. Selecting multiple paths enables BGP to load-balance traffic across multiple links.

A path is considered a BGP equal-cost path (and is used for forwarding) if the BGP path selection process performs a tie-break after comparing the IGP cost to the next-hop. By default, all paths with the same neighboring AS, learned by a multipath-enabled BGP neighbor are considered in the multipath selection process.

BGP typically selects only one best path for each prefix and installs that route in the forwarding table. When BGP multipath is enabled, the device selects multiple equal-cost BGP paths to reach a given destination, and all these paths are installed in the forwarding table. BGP advertises only the active path to its neighbors, unless add-path is in use.

The Junos OS BGP multipath feature supports the following applications:

- Load balancing across multiple links between two routing devices belonging to different autonomous systems (ASs)
- Load balancing across a common subnet or multiple subnets to different routing devices belonging to the same peer AS
- Load balancing across multiple links between two routing devices belonging to different external confederation peers
- Load balancing across a common subnet or multiple subnets to different routing devices belonging to external confederation peers

In a common scenario for load balancing, a customer is multihomed to multiple routers or switches in a point of presence (POP). The default behavior is to send all traffic across only one of the available links. Load balancing causes traffic to use two or more of the links.

BGP multipath does not apply to paths that share the same MED-plus-IGP cost, yet differ in IGP cost. Multipath path selection is based on the IGP cost metric, even if two paths have the same MED-plus-IGP cost.

Starting in Junos OS Release 18.1R1 BGP multipath is supported globally at `[edit protocols bgp]` hierarchy level. You can selectively disable multipath on some BGP groups and neighbors. Include `disable` at `[edit protocols bgp group group-name multipath]` hierarchy level to disable multipath option for a group or a specific BGP neighbor.

Starting in Junos OS Release 18.1R1, you can defer multipath calculation until all BGP routes are received. When multipath is enabled, BGP inserts the route into the multipath queue each time a new route is added or whenever an existing route changes. When multiple paths are received through BGP add-path feature, BGP might calculate one multipath route multiple times. Multipath calculation slows down the RIB (also known as the routing table) learning rate. To speed up RIB learning, multipath calculation can be either deferred until the BGP routes are received or you can lower the priority of the multipath build job as per your requirements until the BGP routes are resolved. To defer the multipath calculation configure `defer-initial-multipath-build` at `[edit protocols bgp]` hierarchy level. Alternatively, you can lower the BGP multipath build job priority using `multipath-build-priority` configuration statement at `[edit protocols bgp]` hierarchy level to speed up RIB learning.

SEE ALSO

> Example: Advertising Multiple BGP Paths to a Destination
>
> *Understanding Per-Packet Load Balancing*

Example: Load Balancing BGP Traffic

IN THIS SECTION

- Requirements | **552**
- Overview | **553**
- Configuration | **555**
- Verification | **558**

This example shows how to configure BGP to select multiple equal-cost external BGP (EBGP) or internal BGP (IBGP) paths as active paths.

Requirements

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol (IGP).
- Configure BGP.
- Configure a routing policy that exports routes (such as direct routes or IGP routes) from the routing table into BGP.

Overview

IN THIS SECTION

- Topology | 554

The following steps show how to configure per-packet load balancing:

1. Define a load-balancing routing policy by including one or more `policy-statement` statements at the `[edit policy-options]` hierarchy level, defining an action of `load-balance per-packet`:

```
policy-statement policy-name {
    from {
        match-conditions;
        route-filter destination-prefix match-type <actions>;
        prefix-list name;
    }
    then {
        load-balance per-packet;
    }
}
```

NOTE: To enable load-balancing among multiple EBGP paths and multiple IBGP paths , include the `multipath` statement globally at the `[edit protocols bgp]` hierarchy level. You cannot enable load-balancing of BGP traffic without including the `multipath` statement globally, or for a BGP group at the `[edit protocols bgp group group-name` hierarchy level, or for specific BGP neighbors at the `[edit protocols bgp group group-name neighbor address]` hierarchy level.

2. Apply the policy to routes exported from the routing table to the forwarding table. To do this, include the `forwarding-table` and `export` statements:

```
forwarding-table {
    export policy-name;
}
```

   You cannot apply the export policy to VRF routing instances.

3. Specify all next hops of that route, if more than one exists, when allocating a label corresponding to a route that is being advertised.

4. Configure the forwarding-options hash key for MPLS to include the IP payload.

NOTE: On some platforms, you can increase the number of paths that are load balanced by using the chassis maximum-ecmp statement.

With this statement, you can change the maximum number of equal-cost load-balanced paths to 32, 64, 128, 256, or 512 (the maximum number varies per platform—see maximum-ecmp.)

The multipath feature is supported on all platforms that support BGP. Some enhancements have been made to QFX platforms:

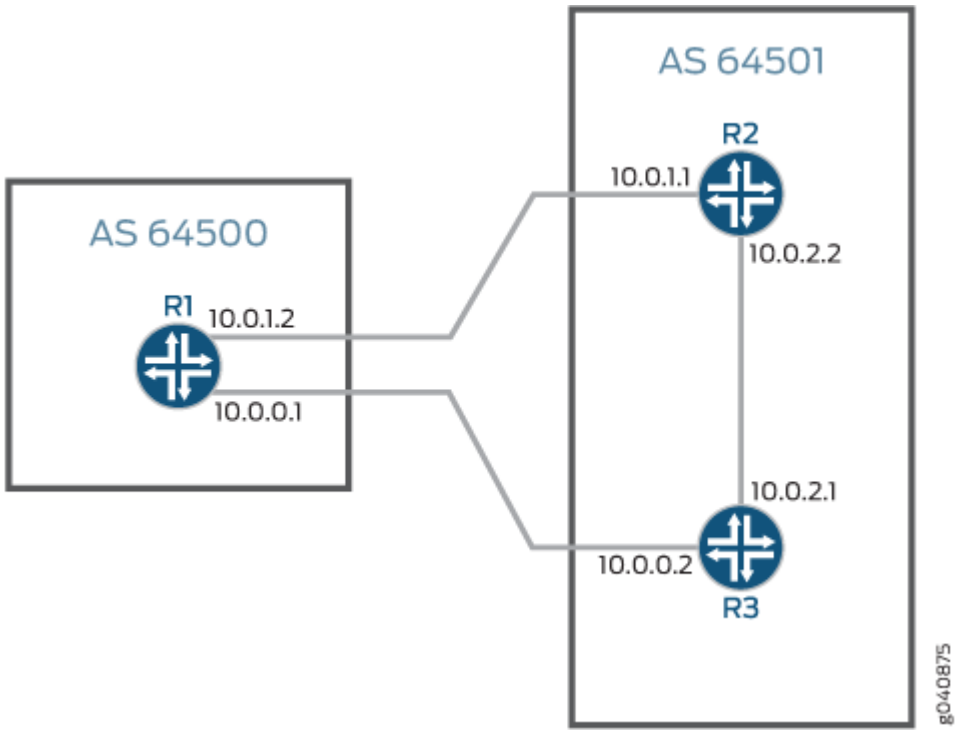
- • Starting with Junos OS Release 19.1R1, you can specify a maximum number of 128 equal-cost paths on QFX10000 switches.
- • Starting with Junos OS Release 19.2R1, you can specify a maximum number of 512 equal-cost paths on QFX10000 switches—see Understanding Configuration of Up to 512 Equal-Cost Paths With Optional Consistent Load Balancing.

In this example, Device R1 is in AS 64500 and is connected to both Device R2 and Device R3, which are in AS 64501. This example shows the configuration on Device R1.

Topology

Figure 42 on page 555 shows the topology used in this example.

Figure 42: BGP Load Balancing



Configuration

IN THIS SECTION

- Procedure | 555

Procedure

CLI Quick Configuration

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

```
set protocols bgp group external type external
set protocols bgp group external peer-as 64501
set protocols bgp group external multipath
```

```
set protocols bgp group external neighbor 10.0.1.1
set protocols bgp group external neighbor 10.0.0.2
set policy-options policy-statement loadbal from route-filter 10.0.0.0/16 orlonger
set policy-options policy-statement loadbal then load-balance per-packet
set routing-options forwarding-table export loadbal
set routing-options autonomous-system 64500
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the BGP group.

```
[edit protocols bgp group external]
user@R1# set type external
user@R1# set peer-as 64501
user@R1# set neighbor 10.0.1.1
user@R1# set neighbor 10.0.0.2
```

2. Enable the BGP group to use multiple paths.

> **NOTE**: To disable the default check requiring that paths accepted by BGP multipath must have the same neighboring autonomous system (AS), include the `multiple-as` option.

```
[edit protocols bgp group external]
user@R1# set multipath
```

3. Configure the load-balancing policy.

```
[edit policy-options policy-statement loadbal]
user@R1# set from route-filter 10.0.0.0/16 orlonger
user@R1# set then load-balance per-packet
```

4. Apply the load-balancing policy.

```
[edit routing-options]
user@R1# set forwarding-table export loadbal
```

5. Configure the local autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 64500
```

Results

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

```
[edit]
user@R1# show protocols
bgp {
    group external {
        type external;
        peer-as 64501;
        multipath;
        neighbor 10.0.1.1;
        neighbor 10.0.0.2;
    }
}
```

```
[edit]
user@R1# show policy-options
policy-statement loadbal {
    from {
        route-filter 10.0.0.0/16 orlonger;
    }
    then {
        load-balance per-packet;
```

```
    }
}
```

```
[edit]
user@R1# show routing-options
autonomous-system 64500;
forwarding-table {
    export loadbal;
}
```

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

Verification

IN THIS SECTION

- Verifying Routes | 558
- Verifying Forwarding | 560

Confirm that the configuration is working properly:

Verifying Routes

Purpose

Verify that routes are learned from both routers in the neighboring AS.

Action

From operational mode, run the `show route` command.

```
user@R1> show route 10.0.2.0
inet.0: 12 destinations, 15 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.2.0/30        *[BGP/170] 03:12:32, localpref 100
                      AS path: 64501 I
```

```
                      to 10.0.1.1 via ge-1/2/0.0
                    > to 10.0.0.2 via ge-1/2/1.0
                    [BGP/170] 03:12:32, localpref 100
                      AS path: 64501 I
                    > to 10.0.1.1 via ge-1/2/0.0
```

```
user@R1> show route 10.0.2.0 detail
inet.0: 12 destinations, 15 routes (12 active, 0 holddown, 0 hidden)
10.0.2.0/30 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 262142
                Next-hop reference count: 3
                Source: 10.0.0.2
                Next hop: 10.0.1.1 via ge-1/2/0.0
                Next hop: 10.0.0.2 via ge-1/2/1.0, selected
                State: <Active Ext>
                Local AS: 64500 Peer AS: 64501
                Age: 3:18:30
                Task: BGP_64501.10.0.0.2+55402
                Announcement bits (1): 2-KRT
                AS path: 64501 I
                Accepted Multipath
                Localpref: 100
                Router ID: 192.168.2.1
         BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 602
                Next-hop reference count: 5
                Source: 10.0.1.1
                Next hop: 10.0.1.1 via ge-1/2/0.0, selected
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Active preferred
                Local AS: 64500 Peer AS: 64501
                Age: 3:18:30
                Task: BGP_64501.10.0.1.1+53135
                AS path: 64501 I
                Accepted
                Localpref: 100
                Router ID: 192.168.3.1
```

Meaning

The active path, denoted with an asterisk (*), has two next hops: 10.0.1.1 and 10.0.0.2 to the 10.0.2.0 destination. The 10.0.1.1 next hop is copied from the inactive path to the active path.

NOTE: The `show route detail` command output designates one gateway as `selected`. This output is potentially confusing in the context of load balancing. The selected gateway is used for many purposes in addition to deciding which gateway to install into the kernel when Junos OS is not performing per-packet load-balancing. For instance, the `ping mpls` command uses the selected gateway when sending packets. Multicast protocols use the selected gateway in some cases to determine the upstream interface. Therefore, even when Junos OS is performing per-packet load-balancing by way of a forwarding-table policy, the selected gateway information is still required for other purposes. It is useful to display the selected gateway for troubleshooting purposes. Additionally, it is possible to use forwarding-table policy to override what is installed into the kernel (for example, by using the `install-nexthop` action). In this case, the next-hop gateway installed in the forwarding table might be a subset of the total gateways displayed in the `show route` command.

Verifying Forwarding

Purpose

Verify that both next hops are installed in the forwarding table.

Action

From operational mode, run the `show route forwarding-table` command.

```
user@R1> show route forwarding-table destination 10.0.2.0
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
10.0.2.0/30        user     0                    ulst 262142     2
                              10.0.1.1           ucst   602     5 ge-1/2/0.0
                              10.0.0.2           ucst   522     6 ge-1/2/1.0
```

SEE ALSO

Understanding BGP Multipath | 551

Understanding External BGP Peering Sessions | 26

Understanding Configuration of Up to 512 Equal-Cost Paths With Optional Consistent Load Balancing

IN THIS SECTION

- Guidelines and Limitations for Configuring from 256 to 512 Equal-Cost Paths, Optionally with Consistent Load Balancing | 561
- Instructions for Configuring Up to 512 ECMP Next Hops, and Optionally Configuring Consistent Load Balancing | 563

You can configure the equal-cost multipath (ECMP) feature with up to 512 paths for external BGP peers. Having the ability to configure up to 512 ECMP next hops allows you to increase the number of direct BGP peer connections with your specified routing device, thus improving latency and optimizing data flow. You can optionally include consistent load balancing in that ECMP configuration. Consistent load balancing ensures that if an ECMP member (that is, a path) fails, only flows flowing through the failed member are redistributed to other active ECMP members. Consistent load balancing also ensures that if an ECMP member is added, redistribution of flows from existing EMCP members to the new ECMP member is minimal.

Guidelines and Limitations for Configuring from 256 to 512 Equal-Cost Paths, Optionally with Consistent Load Balancing

- The feature applies only to single-hop external BGP peers. (This feature does not apply to MPLS routes.)
- The device's routing process (RPD) must support 64-bit mode; 32-bit RPD is not supported.
- The feature applies only to unicast traffic.
- Traffic distribution might not be even across all group members—it depends on the traffic pattern and on the organization of the hashing flow set table in hardware. Consistent hashing *minimizes* remapping of flows to destination links when members are added to or deleted from the group.

- If you configure `set forwarding-options enhanced-hash-key` with one of the options `hash-mode`, `inet`, `inet6`, or `layer2`, some flows might change destination links, because the new hash parameters might generate new hash indexes for the flows, resulting in new destination links.
- To achieve the best-possible hashing accuracy, this feature uses a *cascaded* topology to implement the next-hop structure for configurations of more than 128 next hops. Hashing accuracy is therefore somewhat lesser than it is for ECMP next-hop configurations of less than 128, which do not require a cascaded topology.
- Existing flows on affected ECMP paths and new flows flowing over those affected ECMP paths might switch paths during local route repair, and traffic skewing might be noticeable. However, any such skewing is corrected during the subsequent global route repair.
- When you increase the `maximum-ecmp` value, consistency hashing is lost during the *next* next-hop-change event for the route prefix.
- If you add a new path to an existing ECMP group, some flows over unaffected paths might move to the newly added path.
- Fast reroute (FRR) might not work with consistent hashing.
- Perfect ECMP-like traffic distribution cannot be achieved. Paths that have more “buckets” than other paths have more traffic flows than paths with fewer buckets (a *bucket* is an entry in the load-balancing table’s distribution list that is mapped to an ECMP member index).
- During network topology change events, consistent hashing is lost for network prefixes in some instances because those prefixes point to a new ECMP next hop that does not have all properties of the prefixes’ previous ECMP next hops.
- If multiple network prefixes point to the same ECMP next hop and one or more of those prefixes is enabled with the `consistent-hash` statement, *all* network prefixes pointing to that same ECMP next hop display consistent–hashing behavior.
- Consistent hashing is supported on the equal-cost BGP routes–based ECMP group only. When other protocols or static routes are configured that have priority over BGP routes, consistent hashing is not supported.
- Consistent hashing might have limitations when the configuration is combined with configurations for the following features, because these features have tunnel terminations or traffic engineering that does not use hashing for selecting paths—GRE tunneling; BUM traffic; EVPN-VXLAN; and MPLS TE, autobandwidth.

Instructions for Configuring Up to 512 ECMP Next Hops, and Optionally Configuring Consistent Load Balancing

When you are ready to configure up to 512 next hops, use the following configuration instructions:

1. Configure the maximum number of ECMP next hops—for example, configure 512 ECMP next hops:

```
[edit]
user@host# set chassis maximum-ecmp 512
```

2. Creating a routing policy and enable per-packet load balancing, thus enabling ECMP globally on the system:

```
[edit]
user@host# set routing-options forwarding-table export load-balancing-policy
user@host# set policy-options policy-statement load-balancing-policy then load-balance per-packet
```

3. Enable resiliency on selected prefixes by creating a separate routing policy to match incoming routes to one or more destination prefixes—for example:

```
[edit]
user@host# set policy-options policy-statement c-hash from route-filter 20.0.0.0/24 orlonger
 user@host# set policy-options policy-statement c-hash then load-balance consistent-hash
```

4. Apply an eBGP import policy (for example, “c-hash”) to the BGP group of external peers:

```
[edit]
user@host# set protocols bgp import c-hash
```

For more detail on configuring equal-cost paths, see "Example: Load Balancing BGP Traffic" on page 552, which appears earlier in this document.

(Optional) For more detail on configuring consistent load balancing (also known as consistent hashing), see "Configuring Consistent Load Balancing for ECMP Groups" on page 686

SEE ALSO

Understanding BGP Multipath | 551

Example: Configuring Single-Hop EBGP Peers to Accept Remote Next Hops

IN THIS SECTION

- Requirements | 564
- Overview | 564
- Configuration | 565
- Verification | 577

This example shows how to configure a single-hop external BGP (EBGP) peer to accept a remote next hop with which it does not share a common subnet.

Requirements

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

Overview

In some situations, it is necessary to configure a single-hop EBGP peer to accept a remote next hop with which it does not share a common subnet. The default behavior is for any next-hop address received from a single-hop EBGP peer that is not recognized as sharing a common subnet to be discarded. The ability to have a single-hop EBGP peer accept a remote next hop to which it is not directly connected also prevents you from having to configure the single-hop EBGP neighbor as a multihop session. When you configure a multihop session in this situation, all next-hop routes learned through this EBGP peer are labeled indirect even when they do share a common subnet. This situation breaks multipath functionality for routes that are recursively resolved over routes that include these next-hop addresses. Configuring the `accept-remote-nexthop` statement allows a single-hop EBGP peer to accept a remote next hop, which restores multipath functionality for routes that are resolved over these next-hop addresses. You can configure this statement at the global, group, and neighbor hierarchy levels for BGP. The statement is also supported on logical systems and the VPN routing and forwarding (VRF) routing instance type. Both the remote next-hop and the EBGP peer must support BGP route refresh as defined

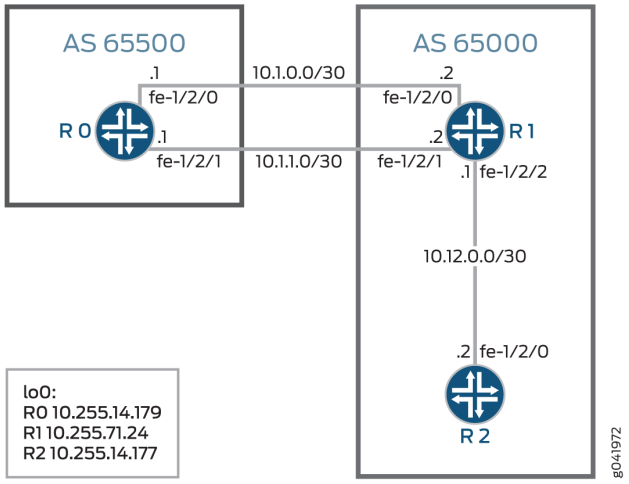
in RFC 2918, *Route Refresh Capability in BGP-4*. If the remote peer does not support BGP route refresh, the session is reset.

A single-hop EBGP peer advertises its own address as the next hop by default. if you want to advertise a different next hop you must define an import routing policy on the EBGP peer. When you enable a single-hop EBGP peer to accept a remote next hop, you can also configure an import routing policy on the EBGP peer. However, a routing policy is not required if you have configured a remote next hop.

This example includes an import routing policy, `agg_route`, that enables a single-hop external BGP peer (Device R1) to accept the remote next-hop 10.1.10.10 for the route to the 10.1.230.0/23 network. At the `[edit protocols bgp]` hierarchy level, the example includes the `import agg_route` statement to apply the policy to the external BGP peer and includes the `accept-remote-nexthop` statement to enable the single-hop EBGP peer to accept the remote next hop.

Figure 43 on page 565 shows the sample topology.

Figure 43: Topology for Accepting a Remote Next Hop



Configuration

IN THIS SECTION

- CLI Quick Configuration | **566**
- Device R0 | **567**
- Configuring Device R1 | **571**

- Configuring Device R2 | 575

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 R0

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.1/30
set interfaces lo0 unit 0 family inet address 10.255.14.179/32
set protocols bgp group ext type external
set protocols bgp group ext export test_route
set protocols bgp group ext export agg_route
set protocols bgp group ext peer-as 65000
set protocols bgp group ext multipath
set protocols bgp group ext neighbor 10.1.0.2
set protocols bgp group ext neighbor 10.1.1.2
set policy-options policy-statement agg_route term 1 from protocol static
set policy-options policy-statement agg_route term 1 from route-filter 10.1.230.0/23 exact
set policy-options policy-statement agg_route term 1 then accept
set policy-options policy-statement test_route term 1 from protocol static
set policy-options policy-statement test_route term 1 from route-filter 10.1.10.10/32 exact
set policy-options policy-statement test_route term 1 then accept
set routing-options static route 10.1.10.10/32 reject
set routing-options static route 10.1.230.0/23 reject
set routing-options autonomous-system 65500
```

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.2/30
set interfaces fe-1/2/2 unit 0 family inet address 10.12.0.1/30
set interfaces lo0 unit 2 family inet address 10.255.71.24/32
set protocols bgp accept-remote-nexthop
set protocols bgp group ext type external
```

```
set protocols bgp group ext import agg_route
set protocols bgp group ext peer-as 65500
set protocols bgp group ext multipath
set protocols bgp group ext neighbor 10.1.0.1
set protocols bgp group ext neighbor 10.1.1.1
set protocols bgp group int type internal
set protocols bgp group int local-address 10.255.71.24
set protocols bgp group int neighbor 10.255.14.177
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set protocols ospf area 0.0.0.0 interface 10.255.71.24
set policy-options policy-statement agg_route term 1 from protocol bgp
set policy-options policy-statement agg_route term 1 from route-filter 10.1.230.0/23 exact
set policy-options policy-statement agg_route term 1 then next-hop 10.1.10.10
set policy-options policy-statement agg_route term 1 then accept
set routing-options autonomous-system 65000
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.12.0.2/30
set interfaces lo0 unit 0 family inet address 10.255.14.177/32
set protocols bgp group int type internal
set protocols bgp group int local-address 10.255.14.177
set protocols bgp group int neighbor 10.255.71.24
set protocols ospf area 0.0.0.0 interface fe-1/2/0.6
set protocols ospf area 0.0.0.0 interface 10.255.14.177
set routing-options autonomous-system 65000
```

Device R0

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 Junos OS CLI User Guide.

To configure Device R0:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 0]
user@R0# set family inet address 10.1.0.1/30
```

```
[edit interfaces fe-1/2/1 unit 0]
user@R0# set family inet address 10.1.1.1/30
[edit interfaces lo0 unit 0]
user@R0# set family inet address 10.255.14.179/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@R0# set type external
user@R0# set peer-as 65000
user@R0# set neighbor 10.1.0.2
user@R0# set neighbor 10.1.1.2
```

3. Enable multipath BGP between Device R0 and Device R1.

```
[edit protocols bgp group ext]
user@R0# set multipath
```

4. Configure static routes to remote networks.
   These routes are not part of the topology. The purpose of these routes is to demonstrate the functionality in this example.

```
[edit routing-options]
user@R0# set static route 10.1.10.10/32 reject
user@R0# set static route 10.1.230.0/23 reject
```

5. Configure routing policies that accept the static routes.

```
[edit policy-options policy-statement agg_route term 1]
user@R0# set from protocol static
user@R0# set from route-filter 10.1.230.0/23 exact
user@R0# set then accept
[edit policy-options policy-statement test_route term 1]
user@R0# set from protocol static
user@R0# set from route-filter 10.1.10.10/32 exact
user@R0# set then accept
```

6. Export the `agg_route` and `test_route` policies from the routing table into BGP.

```
[edit protocols bgp group ext]
user@R0# set export test_route
user@R0# set export agg_route
```

7. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R0# set autonomous-system 65500
```

Results

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

```
user@R0# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.1.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.14.179/32;
        }
```

```
    }
}
```

```
user@R0# show policy-options
policy-statement agg_route {
    term 0 {
        from {
            protocol static;
            route-filter 10.1.230.0/23 exact;
        }
        then accept;
    }
}
policy-statement test_route {
    term 1 {
        from {
            protocol static;
            route-filter 10.1.10.10/32 exact;
        }
        then accept;
    }
}
```

```
user@R0# show protocols
bgp {
    group ext {
        type external;
        export [ test_route agg_route ];
        peer-as 65000;
        multipath;
        neighbor 10.1.0.2;
        neighbor 10.1.1.2;
    }
}
```

```
user@R0# show routing-options
static {
    route 10.1.10.10/32 reject;
    route 10.1.230.0/23 reject;
```

```
}
autonomous-system 65500;
```

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

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 0]
user@R1# set family inet address 10.1.0.2/30
[edit interfaces fe-1/2/1 unit 0]
user@R1# set family inet address 10.1.1.2/30
[edit interfaces fe-1/2/2 unit 0]
user@R1# set family inet address 10.12.0.1/30
[edit interfaces lo0 unit 0]
user@R1# set family inet address 10.255.71.24/32
```

2. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/1.0
user@R1# set interface 10.255.71.24
```

3. Enable Device R1 to accept the remote next hop.

```
[edit protocols bgp]
user@R1# set accept-remote-nexthop
```

4. Configure IBGP.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 10.255.71.24
user@R1# set neighbor 10.255.14.177
```

5. Configure EBGP.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set peer-as 65500
user@R1# set neighbor 10.1.0.1
user@R1# set neighbor 10.1.1.1
```

6. Enable multipath BGP between Device R0 and Device R1.

```
[edit protocols bgp group ext]
user@R1# set multipath
```

7. Configure a routing policy that enables a single-hop external BGP peer (Device R1) to accept the remote next-hop 10.1.10.10 for the route to the 10.1.230.0/23 network.

```
[edit policy-options policy-statement agg_route term 1]
user@R1# set from protocol bgp
user@R1# set from route-filter 10.1.230.0/23 exact
user@R1# set then next-hop 10.1.10.10
user@R1# set then accept
```

8. Import the `agg_route` policy into the routing table on Device R1.

```
[edit protocols bgp group ext]
user@R1# set import agg_route
```

9. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 65000
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.1.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.1.2/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        family inet {
            address 10.12.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.71.24/32;
        }
```

```
    }
}
```

```
user@R1# show policy-options
policy-statement agg_route {
    term 1 {
        from {
            protocol bgp;
            route-filter 10.1.230.0/23 exact;
        }
        then {
            next-hop 10.1.10.10;
            accept;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    accept-remote-nexthop;
    group ext {
        type external;
        import agg_route;
        peer-as 65500;
        multipath;
        neighbor 10.1.0.1;
        neighbor 10.1.1.1;
    }
    group int {
        type internal;
        local-address 10.255.71.24;
        neighbor 10.255.14.177;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.0;
        interface 10.255.71.24;
```

```
    }
}
```

```
user@R1# show routing-options
autonomous-system 65000;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0 unit 0]
user@R2# set family inet address 10.12.0.2/30
[edit  interfaces lo0 unit 0]
user@R2# set family inet address 10.255.14.177/32
```

2. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface fe-1/2/0.0
user@R2# set interface 10.255.14.177
```

3. Configure IBGP.

```
[edit protocols bgp group int]
user@R2# set type internal
user@R2# set local-address 10.255.14.177
user@R2# set neighbor 10.255.71.24
```

4. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 65000
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.12.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.14.177/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group int {
        type internal;
        local-address 10.255.14.177;
        neighbor 10.255.71.24;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.0;
        interface 10.255.14.177;
```

```
    }
}
```

```
user@R2# show routing-options
autonomous-system 65000;
```

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

Verification

IN THIS SECTION

- Verifying That the Multipath Route with the Indirect Next Hop Is in the Routing Table | 577
- Deactivating and Reactivating the accept-remote-nexthop Statement | 579

Confirm that the configuration is working properly.

Verifying That the Multipath Route with the Indirect Next Hop Is in the Routing Table

Purpose

Verify that Device R1 has a route to the 10.1.230.0/23 network.

Action

From operational mode, enter the `show route 10.1.230.0 extensive` command.

```
user@R1> show route 10.1.230.0 extensive
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
Restart Complete
10.1.230.0/23 (2 entries, 1 announced)
TSI:
KRT in-kernel 10.1.230.0/23 -> {indirect(262142)}
Page 0 idx 1 Type 1 val 9168f6c
    Nexthop: 10.1.10.10
    Localpref: 100
    AS path: [65000] 65500 I
```

```
    Communities:
Path 10.1.230.0 from 10.1.0.1 Vector len 4.  Val: 1
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x90c44d8
                Next-hop reference count: 4
                Source: 10.1.0.1
                Next hop type: Router, Next hop index: 262143
                Next hop: 10.1.0.1 via fe-1/2/0.0, selected
                Next hop: 10.1.1.1 via fe-1/2/2.0
                Protocol next hop: 10.1.10.10
                Indirect next hop: 91c0000 262142
                State: <Active Ext>
                Local AS: 65000 Peer AS: 65500
                Age: 2:55:31    Metric2: 0
                Task: BGP_65500.10.1.0.1+64631
                Announcement bits (3): 2-KRT 3-BGP_RT_Background 4-Resolve tree 1
                AS path: 65500 I
                Accepted Multipath
                Localpref: 100
                Router ID: 10.255.14.179
                Indirect next hops: 1
                        Protocol next hop: 10.1.10.10
                        Indirect next hop: 91c0000 262142
                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 10.1.0.1 via fe-1/2/0.0
                                Next hop: 10.1.1.1 via fe-1/2/2.0
                        10.1.10.10/32 Originating RIB: inet.0
                          Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.1.0.1 via fe-1/2/0.0
                                Nexthop: 10.1.1.1 via fe-1/2/2.0
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x90c44d8
                Next-hop reference count: 4
                Source: 10.1.1.1
                Next hop type: Router, Next hop index: 262143
                Next hop: 10.1.0.1 via fe-1/2/0.0, selected
                Next hop: 10.1.1.1 via fe-1/2/2.0
                Protocol next hop: 10.1.10.10
                Indirect next hop: 91c0000 262142
```

```
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Update source
                Local AS: 65000 Peer AS: 65500
                Age: 2:55:27    Metric2: 0
                Task: BGP_65500.10.1.1.1+53260
                AS path: 65500 I
                Accepted
                Localpref: 100
                Router ID: 10.255.14.179
                Indirect next hops: 1
                        Protocol next hop: 10.1.10.10
                        Indirect next hop: 91c0000 262142
                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 10.1.0.1 via fe-1/2/0.0
                                Next hop: 10.1.1.1 via fe-1/2/2.0
                        10.1.10.10/32 Originating RIB: inet.0
                          Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.1.0.1 via fe-1/2/0.0
                                Nexthop: 10.1.1.1 via fe-1/2/2.0
```

Meaning

The output shows that Device R1 has a route to the 10.1.230.0 network with the multipath feature enabled (`Accepted Multipath`). The output also shows that the route has an indirect next hop of 10.1.10.10.

Deactivating and Reactivating the accept-remote-nexthop Statement

Purpose

Make sure that the multipath route with the indirect next hop is removed from the routing table when you deactivate the `accept-remote-nexthop` statement.

Action

1. From configuration mode, enter the `deactivate protocols bgp accept-remote-nexthop` command.

```
user@R1# deactivate protocols bgp accept-remote-nexthop
user@R1# commit
```

2. From operational mode, enter the `show route 10.1.230.0` command.

```
user@R1> show route 10.1.230.0
```

3. From configuration mode, reactivate the statement by entering the `activate protocols bgp accept-remote-nexthop` command.

```
user@R1# activate protocols bgp accept-remote-nexthop
user@R1# commit
```

4. From operational mode, reenter the `show route 10.1.230.0` command.

```
user@R1> show route 10.1.230.0

inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.1.230.0/23       *[BGP/170] 03:13:19, localpref 100
                      AS path: 65500 I
                    > to 10.1.0.1 via fe-1/2/0.0
                      to 10.1.1.1 via fe-1/2/2.0
                    [BGP/170] 03:13:15, localpref 100, from 10.1.1.1
                      AS path: 65500 I
                    > to 10.1.0.1 via fe-1/2/0.0
                      to 10.1.1.1 via fe-1/2/2.0
```

Meaning

When the `accept-remote-nexthop` statement is deactivated, the multipath route to the 10.1.230.0 network is removed from the routing table .

SEE ALSO

Understanding BGP Path Selection | 12

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

Understanding Load Balancing for BGP Traffic with Unequal Bandwidth Allocated to the Paths

The multipath option removes the tiebreakers from the active route decision process, thereby allowing otherwise equal cost BGP routes learned from multiple sources to be installed into the forwarding table. However, when the available paths are not equal cost, you may wish to load balance the traffic asymmetrically.

Once multiple next hops are installed in the forwarding table, a specific forwarding next hop is selected by the Junos OS per-prefix load-balancing algorithm. This process hashes against a packet's source and destination addresses to deterministically map the prefix pairing onto one of the available next hops. Per-prefix mapping works best when the hash function is presented with a large number of prefixes, such as might occur on an Internet peering exchange, and it serves to prevent packet reordering among pairs of communicating nodes.

An enterprise network normally wants to alter the default behavior to evoke a *per-packet* load-balancing algorithm. Per-packet is emphasized here because its use is a misnomer that stems from the historic behavior of the original Internet Processor ASIC. In reality, current Juniper Networks routers support per-prefix (default) and per-flow load balancing. The latter involves hashing against various Layer 3 and Layer 4 headers, including portions of the source address, destination address, transport protocol, incoming interface, and application ports. The effect is that now individual flows are hashed to a specific next hop, resulting in a more even distribution across available next hops, especially when routing between fewer source and destination pairs.

With per-packet load balancing, packets comprising a communication stream between two endpoints might be resequenced, but packets within individual flows maintain correct sequencing. Whether you opt for per-prefix or per-packet load balancing, asymmetry of access links can present a technical challenge. Either way, the prefixes or flows that are mapped to, for example, a T1 link will exhibit degraded performance when compared to those flows that map to, for example, a Fast Ethernet access link. Worse yet, with heavy traffic loads, any attempt at equal load balancing is likely to result in total saturation of the T1 link and session disruption stemming from packet loss.

Fortunately, the Juniper Networks BGP implementation supports the notion of a bandwidth community. This extended community encodes the bandwidth of a given next hop, and when combined with multipath, the load-balancing algorithm distributes flows across the set of next hops proportional to their relative bandwidths. Put another way, if you have a 10-Mbps and a 1-Mbps next hop, on average nine flows will map to the high-speed next hop for every one that uses the low speed.

Use of BGP bandwidth community is supported only with per-packet load balancing.

The configuration task has two parts:

- Configure the external BGP (EBGP) peering sessions, enable multipath, and define an import policy to tag routes with a bandwidth community that reflects link speed.

- Enable per-packet (really per-flow) load balancing for optimal distribution of traffic.

SEE ALSO

> *Understanding Per-Packet Load Balancing*

Example: Load Balancing BGP Traffic with Unequal Bandwidth Allocated to the Paths

IN THIS SECTION

- Requirements | **582**
- Overview | **583**
- Configuration | **585**
- Verification | **592**

This example shows how to configure BGP to select multiple unequal-cost paths as active paths.

BGP communities can help you control routing policy. An example of a good use for BGP communities is unequal load balancing. When an autonomous system border router (ASBR) receives routes from directly connected external BGP (EBGP) neighbors, the ASBR then advertises those routes to internal neighbors, using IBGP advertisements. In the IBGP adverisements, you can attach the link-bandwidth community to communicate the bandwidth of the advertised external link. This is useful when multiple external links are available, and you want to do unequal load balancing over the links. You configure the link-bandwidth extended community on all ingress links of the AS. The bandwidth information in the link-bandwidth extended community is based on the configured bandwidth of the EBGP link. It is not based on the amount of traffic on the link. Junos OS supports BGP link-bandwidth and multipath load balancing, as described in Internet draft draft-ietf-idr-link-bandwidth-06, *BGP Link Bandwidth Extended Community*.

Requirements

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol (IGP).

- Configure BGP.
- Configure a routing policy that exports routes (such as direct routes or IGP routes) from the routing table into BGP.

Overview

IN THIS SECTION

- Topology | 584

In this example, Device R1 is in AS 64500 and is connected to both Device R2 and Device R3, which are in AS 64501.

The example uses the bandwidth extended community.

By default, when BGP multipath is used, traffic is distributed equally among the several paths calculated. The bandwidth extended community allows an additional attribute to be added to BGP paths, thus allowing the traffic to be distributed unequally. The primary application is a scenario where multiple external paths exist for a given network with asymmetric bandwidth capabilities. In such a scenario, you can tag routes received with the bandwidth extended community. When BGP multipath (internal or external) operates among routes that contain the bandwidth attribute, the forwarding engine can unequally distribute traffic according to the bandwidth corresponding to each path.

When BGP has several candidate paths available for multipath purposes, BGP does not perform unequal cost load balancing according to the bandwidth community unless all candidate paths have this attribute.

The applicability of the bandwidth extended community is limited by the restrictions under which BGP multipath accepts multiple paths for consideration. Explicitly, the IGP distance, as far as BGP is concerned, between the router performing load balancing and the multiple exit points needs to be the same. This can be achieved by using a full mesh of label-switched paths (LSPs) that do not track the corresponding IGP metric. However, in a network in which the propagation delay of circuits is significant (for example, if long-haul circuits are present), it is often valuable to take into account the delay characteristics of different paths.

Configure the bandwidth community as follows:

```
[edit policy-options]
user@host# set community members bandwidth:[1-65535]:[0-4294967295]
```

The first 16-bit number represents the local autonomous system. The second 32-bit number represents the link bandwidth in bytes per second.

For example:

```
[edit policy-options]
user@host# show
community bw-t1 members bandwidth:10458:193000;
community bw-t3 members bandwidth:10458:5592000;
community bw-oc3 members bandwidth:10458:19440000;
```

Where 10458 is the local AS number. The values correspond to the bandwidth of the T1, T3, and OC-3 paths in bytes per second. The value specified as the bandwidth value does not need to correspond to the actual bandwidth of a specific interface. The balance factors used are calculated as a function of the total bandwidth specified. To tag a route with this extended community, define a policy statement, as follows:

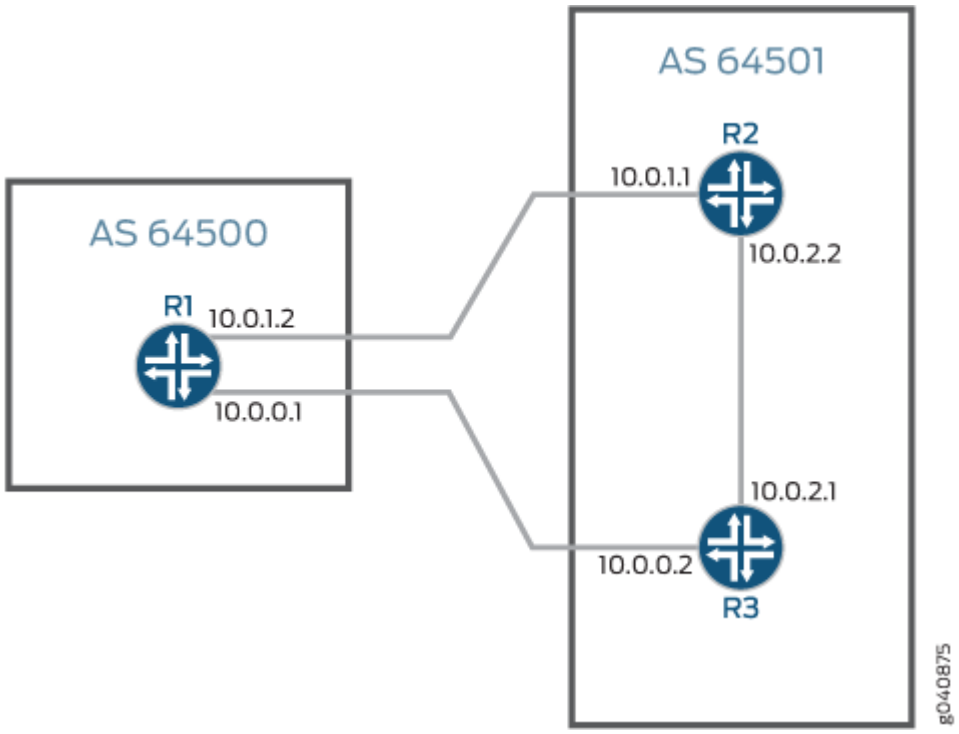
```
[edit policy-options]
user@host# show
policy-statement link-bw-t1 {
    then {
        community set bw-t1;
    }
    accept;
}
```

Apply this as an import policy on the BGP peering sessions facing the asymmetrical bandwidth links. Although in theory the community attribute can be added or removed at any point in the network, in the scenario described above, applying the community as an import policy in the EBGP peering session facing the external link allows for that attribute to influence the local multipath decision, and is potentially easier to manage.

Topology

Figure 44 on page 585 shows the topology used in this example.

Figure 44: BGP Load Balancing



"CLI Quick Configuration" on page 585 shows the configuration for all of the devices in Figure 44 on page 585. The section"No Link Title" on page 588 describes the steps on Device R1.

Configuration

IN THIS SECTION

- Procedure | **585**

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-1/2/0 unit 0 description R1->R3
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/2/1 unit 0 description R1->R2
set interfaces ge-1/2/1 unit 0 family inet address 10.0.1.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group external type external
set protocols bgp group external import bw-dis
set protocols bgp group external peer-as 64501
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.1.1
set protocols bgp group external neighbor 10.0.0.2
set policy-options policy-statement bw-dis term a from protocol bgp
set policy-options policy-statement bw-dis term a from neighbor 10.0.1.1
set policy-options policy-statement bw-dis term a then community add bw-high
set policy-options policy-statement bw-dis term a then accept
set policy-options policy-statement bw-dis term b from protocol bgp
set policy-options policy-statement bw-dis term b from neighbor 10.0.0.2
set policy-options policy-statement bw-dis term b then community add bw-low
set policy-options policy-statement bw-dis term b then accept
set policy-options policy-statement loadbal from route-filter 10.0.0.0/16 orlonger
set policy-options policy-statement loadbal then load-balance per-packet
set policy-options community bw-high members bandwidth:65000:60000000
set policy-options community bw-low members bandwidth:65000:40000000
set routing-options autonomous-system 64500
set routing-options forwarding-table export loadbal
```

Device R2

```
set interfaces ge-1/2/0 unit 0 description R2->R1
set interfaces ge-1/2/0 unit 0 family inet address 10.0.1.1/30
set interfaces ge-1/2/1 unit 0 description R2->R3
set interfaces ge-1/2/1 unit 0 family inet address 10.0.2.2/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family iso address 49.0001.1921.6800.0002.00
set protocols bgp group external type external
set protocols bgp group external export bgp-default
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 64500
```

```
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.1.2
set protocols isis interface ge-1/2/1.0
set protocols isis interface lo0.0
set policy-options policy-statement bgp-default from protocol static
set policy-options policy-statement bgp-default from route-filter 172.16.0.0/16 exact
set policy-options policy-statement bgp-default then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options static route 172.16.0.0/16 discard
set routing-options static route 172.16.0.0/16 no-install
set routing-options autonomous-system 64501
```

Device R3

```
set interfaces ge-1/2/0 unit 0 description R3->R2
set interfaces ge-1/2/0 unit 0 family inet address 10.0.2.1/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/1 unit 0 description R3->R1
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family iso address 49.0001.1921.6800.0003.00
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external export bgp-default
set protocols bgp group external peer-as 64500
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.0.1
set protocols isis interface ge-1/2/0.0
set protocols isis interface lo0.0
set policy-options policy-statement bgp-default from protocol static
set policy-options policy-statement bgp-default from route-filter 172.16.0.0/16 exact
set policy-options policy-statement bgp-default then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options static route 172.16.0.0/16 discard
set routing-options static route 172.16.0.0/16 no-install
set routing-options autonomous-system 64501
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the interfaces.

```
user@R1# set ge-1/2/0 unit 0 description R1->R3
user@R1# set ge-1/2/0 unit 0 family inet address 10.0.0.1/30
user@R1# set ge-1/2/1 unit 0 description R1->R2
user@R1# set ge-1/2/1 unit 0 family inet address 10.0.1.2/30
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the BGP group.

```
[edit protocols bgp group external]
user@R1# set type external
user@R1# set import bw-dis
user@R1# set peer-as 64501
user@R1# set neighbor 10.0.1.1
user@R1# set neighbor 10.0.0.2
```

3. Enable the BGP group to use multiple paths.

> **NOTE**: To disable the default check requiring that paths accepted by BGP multipath must have the same neighboring autonomous system (AS), include the `multiple-as` option. Use the `multiple-as` option if the neighbors are in different ASs.

```
[edit protocols bgp group external]
user@R1# set multipath
```

4. Configure the load-balancing policy.

```
[edit policy-options policy-statement loadbal]
user@R1# set from route-filter 10.0.0.0/16 orlonger
user@R1# set then load-balance per-packet
```

5. Apply the load-balancing policy.

```
[edit routing-options]
user@R1# set forwarding-table export loadbal
```

6. Configure the BGP community members.

   This example assumes a bandwidth of 1 Gbps and allocates 60 percent to bw-high and 40 percent to bw-low. The reference bandwidth does not need to be the same as the link bandwidth.

```
[edit policy-options]
user@R1# set community bw-high members bandwidth:65000:60000000
user@R1# set community bw-low members bandwidth:65000:40000000
```

7. Configure the bandwidth distribution policy.

```
[edit policy-options bw-dis]
user@R1# set term a from protocol bgp
user@R1# set term a from neighbor 10.0.1.1
user@R1# set term a then community add bw-high
user@R1# set term a then accept
user@R1# set term b from protocol bgp
user@R1# set term b from neighbor 10.0.0.2
user@R1# set term b then community add bw-low
user@R1# set term b then accept
```

8. Configure the local autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 64500
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
ge-1/2/0 {
    unit 0 {
        description R1->R3;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
ge-1/2/1 {
    unit 0 {
        description R1->R2;
        family inet {
            address 10.0.1.2/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group external {
        type external;
        import bw-dis;
        peer-as 64501;
        multipath;
        neighbor 10.0.1.1;
        neighbor 10.0.0.2;
```

```
    }
}
```

```
user@R1# show policy-options
policy-statement bw-dis {
    term a {
        from {
            protocol bgp;
            neighbor 10.0.1.1;
        }
        then {
            community add bw-high;
            accept;
        }
    }
    term b {
        from {
            protocol bgp;
            neighbor 10.0.0.2;
        }
        then {
            community add bw-low;
            accept;
        }
    }
}
policy-statement loadbal {
    from {
        route-filter 10.0.0.0/16 orlonger;
    }
    then {
        load-balance per-packet;
    }
}
community bw-high members bandwidth:65000:60000000;
community bw-low members bandwidth:65000:40000000;
```

```
user@R1# show routing-options
autonomous-system 64500;
forwarding-table {
```

```
    export loadbal;
}
```

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

Verification

IN THIS SECTION

- Verifying Routes | 592

Confirm that the configuration is working properly:

Verifying Routes

Purpose

Verify that both routes are selected and that the next hops on the routes show a 60%/40% balance.

Action

From operational mode, run the `show route protocol bgp detail` command.

```
user@R1> show route 172.16/16 protocol bgp detail
inet.0: 9 destinations, 13 routes (9 active, 0 holddown, 0 hidden)
172.16.0.0/16 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 262143
                Address: 0x93fc078
                Next-hop reference count: 3
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via ge-1/2/0.0 balance 40%
                Next hop: 10.0.1.1 via ge-1/2/1.0 balance 60%, selected
                State: **Active Ext>
                Local AS: 64500 Peer AS: 64501
                Age: 3:22:55
                Task: BGP_64501.10.0.0.2+55344
                Announcement bits (1): 0-KRT
                AS path: 64501 I
```

```
                Communities: bandwidth:65000:40000000
                Accepted Multipath
                Localpref: 100
                Router ID: 192.168.0.3
         BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 658
                Address: 0x9260520
                Next-hop reference count: 4
                Source: 10.0.1.1
                Next hop: 10.0.1.1 via ge-1/2/1.0, selected
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Active preferred
                Local AS: 64500 Peer AS: 64501
                Age: 3:22:55
                Task: BGP_65001.10.0.1.1+62586
                AS path: 64501 I
                Communities: bandwidth:65000:60000000
                Accepted MultipathContrib
                Localpref: 100
                Router ID: 192.168.0.2
```

```
user@R1> show route 10.0.2.0 protocol bgp detail
inet.0: 9 destinations, 13 routes (9 active, 0 holddown, 0 hidden)
10.0.2.0/30 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 262143
                Address: 0x93fc078
                Next-hop reference count: 3
                Source: 10.0.1.1
                Next hop: 10.0.0.2 via ge-1/2/0.0 balance 40%
                Next hop: 10.0.1.1 via ge-1/2/1.0 balance 60%, selected
                State: <Active Ext>
                Local AS: 64500 Peer AS: 64501
                Age: 3:36:37
                Task: BGP_65001.10.0.1.1+62586
                Announcement bits (1): 0-KRT
                AS path: 64501 I
                Communities: bandwidth:65000:60000000
                Accepted Multipath
                Localpref: 100
                Router ID: 192.168.0.2
```

```
          BGP    Preference: 170/-101
                 Next hop type: Router, Next hop index: 657
                 Address: 0x92604d8
                 Next-hop reference count: 4
                 Source: 10.0.0.2
                 Next hop: 10.0.0.2 via ge-1/2/0.0, selected
                 State: <NotBest Ext>
                 Inactive reason: Not Best in its group - Active preferred
                 Local AS: 64500 Peer AS: 65001
                 Age: 3:36:36
                 Task: BGP_65001.10.0.0.2+55344
                 AS path: 64501 I
                 Communities: bandwidth:65000:40000000
                 Accepted MultipathContrib
                 Localpref: 100
                 Router ID: 192.168.0.3
```

Meaning

The active path, denoted with an asterisk (*), has two next hops: 10.0.1.1 and 10.0.0.2 to the 172.16/16 destination.

Likewise, the active path, denoted with an asterisk (*), has two next hops: 10.0.1.1 and 10.0.0.2 to the 10.0.2.0 destination.

In both cases, the 10.0.1.1 next hop is copied from the inactive path to the active path.

The balance of 40 percent and 60 percent is shown in the `show route` output. This indicates that traffic is being distributed between two next hops and that 60 percent of the traffic is following the first path, while 40 percent is following the second path.

SEE ALSO

Understanding BGP Multipath | 551

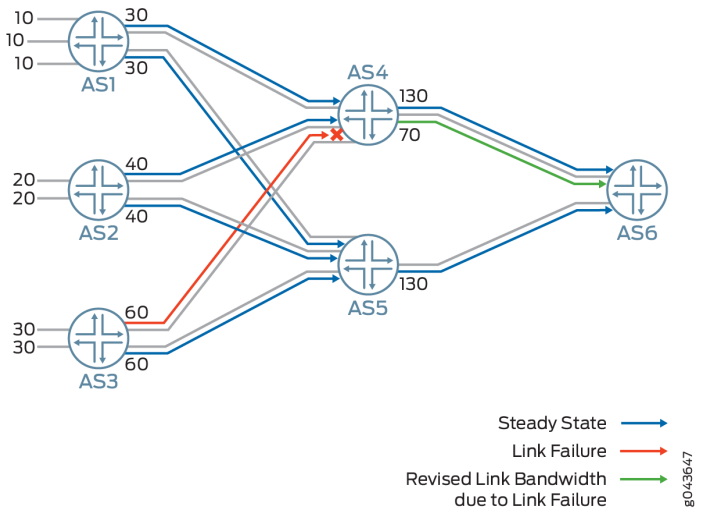
Advertising Aggregate Bandwidth Across External BGP Links for Load Balancing Overview

A BGP peer that receives multiple paths from its internal peers load balances traffic among these paths. In releases prior to Junos OS release 17.4, a BGP speaker receiving multiple paths from its internal peers

advertised only the link bandwidth associated with the active route. BGP uses the link bandwidth extended community, to advertise the aggregated bandwidth of multiple routes across external links. BGP calculates the aggregate bandwidth of multipaths that have unequal bandwidth allocation and advertises the aggregated bandwidth to external BGP peers. A threshold to the aggregate bandwidth can be configured to restrict the bandwidth usage of a BGP group. Both IPv4 and IPv6 routes including anycast addresses support aggregate bandwidth.

To advertise aggregate bandwidth of multipath routes and to set a maximum threshold, configure a policy with `aggregate-bandwidth` and `limit-bandwidth` actions at the [edit policy-options policy-statement *name* then] hierarchy level.

Figure 45: Advertising Aggregate Bandwidth Across External BGP Links for Load Balancing



In Figure 45 on page 595, autonomous system 1 (AS1) aggregates the bandwidth of its 3 multipath routes to a remote prefix and advertises it to autonomous system 4 (AS4) with a bandwidth of 30 using the link bandwidth extended community. In case of a link failure between AS3 and AS4, AS4 subtracts 60 from the bandwidth it advertises to AS6 and modifies the bandwidth it was advertising from 130 to 70.

When one of the multipath links fails, BGP readvertises the route with the bandwidth of the failed link subtracted from the outgoing link bandwidth community. If the aggregate link bandwidth is found to exceed the configured limit, the advertised aggregate bandwidth is truncated to the configured link bandwidth limit between the two peers.

When a BGP peer propagates multipath routes configured with an aggregate bandwidth community, a new link bandwidth community is added with the sum of the bandwidth from the incoming bandwidth communities or that prefix. The available link bandwidth is dynamically derived from interface speed. The link bandwidth is sent as a transitive extended community.

BGP can communicate link speeds to remote peers, enabling better optimization of traffic distribution for load balancing. A BGP group can send the link-bandwidth non-transitive extended community over an EBGP session for originated or received and readvertised link-bandwidth extended communities. To configure the non-transitive link bandwidth extended community:

- Include the `bandwidth-non-transitive:value` in the export policy at the `[edit policy-options community name members community-ids]` hierarchy level.
- Include the `send-non-transitive-link-bandwidth` option at the `[edit protocols bgp No Link Title group-name]` hierarchy level to send non-transitive-link-bandwidth-extended-community towards EBGP neighbors only for link-bandwidth communities.

To enable the device to automatically detect and attach the link-bandwidth community on a route at import, include the No Link Title statement at the `[edit protocols bgp group link-bandwidth]` hierarchy level. This feature facilitates the integration of devices with different transmission speeds within the network, enabling efficient traffic distribution based on link speed.

SEE ALSO

> *policy-statement*
>
> No Link Title

Example: Configuring a Policy to Advertise Aggregate Bandwidth Across External BGP Links for Load Balancing

IN THIS SECTION

- Requirements | **597**
- Overview | **597**
- Configuration | **598**
- Verification | **607**

This example shows how to configure a policy to advertise aggregate bandwidth across External BGP links for load balancing and to specify a threshold for the configured aggregate bandwidth. BGP adds up the available link bandwidth of multipaths and calculates the aggregated bandwidth. In case of a link failure, the aggregated bandwidth is adjusted to reflect the current status of the available bandwidth.

Requirements

This example uses the following hardware and software components:

- Four routers with load balancing capability
- Junos OS Release 17.4 or later running on all the devices

Overview

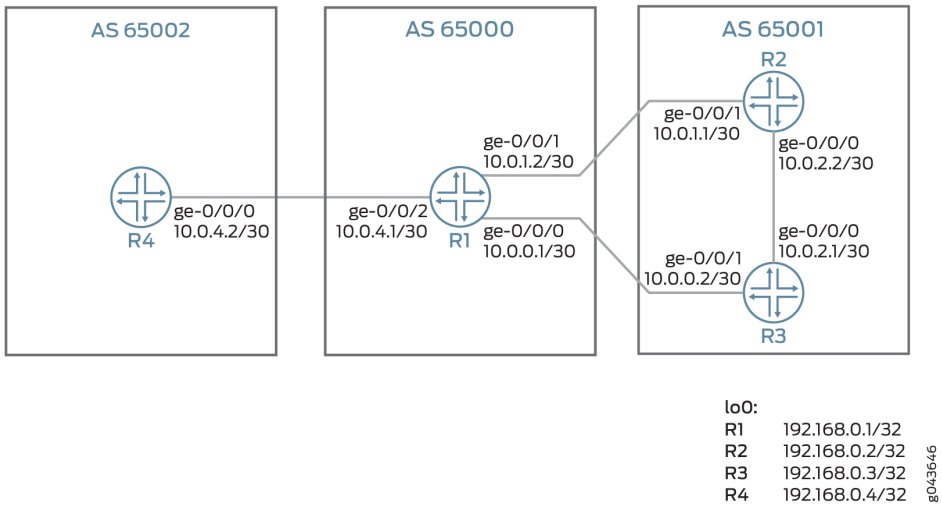
IN THIS SECTION

- Topology | **598**

Starting in Junos OS Release 17.4R1, a BGP speaker that receives multiple paths from its internal peers load balances traffic among these paths. In earlier Junos OS releases, a BGP speaker receiving multiple paths from its internal peers advertised only the link bandwidth associated with the active route. BGP uses a new link bandwidth extended community with the aggregated bandwidth to tag multipaths and advertises the aggregated bandwidth for these multiple routes across its DMZ link. To advertise aggregated multiple routes, configure a policy with `aggregate-bandwidth` and `limit bandwidth` actions at the [edit policy-options policy-statement *name* then] hierarchy level.

Topology

Figure 46: Configuring a Policy to Advertise Aggregate Bandwidth Across External BGP Links for Load Balancing



In Figure 46 on page 598, Router R1 load balances traffic to a remote destination through next-hop 10.0.1.1 in Router R2 at 60,000,000 bytes per second and through 10.0.0.2 in Router R3 at 40,000,000 bytes per second. Router R1 advertises destination 10.0.2.0 to Router R4. Router R1 calculates the aggregate of the available bandwidth, which is 10000000 bytes per second. However, a policy configured on Router R1 sets the threshold for the aggregate bandwidth to 80,000,000 bytes per second. Therefore, R1 advertises 80,000,000 bytes per second instead of the 10,000,000 bytes per second.

NOTE: If one of the multipath links goes down, then the bandwidth of the failed link is not added to the aggregate bandwidth that is advertised to BGP neighbors.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 599
- Configuring Routers, Starting with R1 | 601
- Results | 604

CLI Quick Configuration

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

Router R1

```
set interfaces ge-0/0/0 unit 0 description R1->R3
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-0/0/1 unit 0 description R1->R2
set interfaces ge-0/0/1 unit 0 family inet address 10.0.1.2/30
set interfaces ge-0/0/2 unit 0 description R1->R4
set interfaces ge-0/0/2 unit 0 family inet address 10.0.4.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set routing-options autonomous-system 65000
set protocols bgp group external type external
set protocols bgp group external import bw-dis
set protocols bgp group external peer-as 65001
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.1.1
set protocols bgp group external neighbor 10.0.0.2
set protocols bgp group external2 type external
set protocols bgp group external2 peer-as 65002
set policy-options policy-statement bw-dis term a from protocol bgp
set policy-options policy-statement bw-dis term a from neighbor 10.0.1.1
set policy-options policy-statement bw-dis term a then community add bw-high
set policy-options policy-statement bw-dis term a then accept
set policy-options policy-statement bw-dis term b from protocol bgp
set policy-options policy-statement bw-dis term b from neighbor 10.0.0.2
set policy-options policy-statement bw-dis term b then community add bw-low
set policy-options policy-statement bw-dis term b then accept
set policy-options policy-statement aggregate_bw_and_limit_capacity then aggregate-bandwidth
set policy-options policy-statement aggregate_bw_and_limit_capacity then limit-bandwidth 80000000
set policy-options policy-statement aggregate_bw_and_limit_capacity then accept
set protocols bgp group external2 neighbor 10.0.4.2 export aggregate_bw_and_limit_capacity
set policy-options policy-statement loadbal from route-filter 10.0.0.0/16 orlonger
set policy-options policy-statement loadbal then load-balance per-packet
set routing-options forwarding-table export loadbal
set policy-options community bw-high members bandwidth:65000:60000000
set policy-options community bw-low members bandwidth:65000:40000000
```

Router R2

```
set interfaces ge-0/0/0 unit 0 description R2->R3
set interfaces ge-0/0/0 unit 0 family inet address 10.0.2.2/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/1 unit 0 description R2->R1
set interfaces ge-0/0/1 unit 0 family inet address 10.0.1.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family iso address 49.0001.1921.6800.0002.00
set routing-options static route 172.16.0.0/16 discard
set routing-options static route 172.16.0.0/16 no-install
set routing-options autonomous-system 65001
set protocols bgp group external type external
set protocols bgp group external export bgp-default
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 65000
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.1.2
set protocols isis interface ge-0/0/0.0
set protocols isis interface lo0.0
set policy-options policy-statement bgp-default from protocol static
set policy-options policy-statement bgp-default from route-filter 172.16.0.0/16 exact
set policy-options policy-statement bgp-default then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
```

Router R3

```
set interfaces ge-0/0/0 description R3->R2
set interfaces ge-0/0/0 unit 0 family inet address 10.0.2.1/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/1 unit 0 description R3->R1
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family iso address 49.0001.1921.6800.0003.00
set routing-options static route 172.16.0.0/16 discard
set routing-options static route 172.16.0.0/16 no-install
set routing-options autonomous-system 65001
set protocols bgp group external type external
set protocols bgp group external export bgp-default
set protocols bgp group external export send-direct
```

```
set protocols bgp group external peer-as 65000
set protocols bgp group external multipath
set protocols bgp group external neighbor 10.0.0.1
set protocols isis interface ge-0/0/0.0
set protocols isis interface lo0.0
set policy-options policy-statement bgp-default from protocol static
set policy-options policy-statement bgp-default from route-filter 172.16.0.0/16 exact
set policy-options policy-statement bgp-default then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
```

Router R4

```
set interfaces ge-0/0/0 unit 0 description R4->R1
set interfaces ge-0/0/0 unit 0 family inet address 10.0.4.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set routing-options autonomous-system 65002
set protocols bgp group external type external
set protocols bgp group external peer-as 65000
set protocols bgp group external neighbor 10.0.4.1
```

Configuring Routers, Starting with R1

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 policy to advertise an aggregated bandwidth to BGP peers (starting with Router R1):

NOTE: Repeat this procedure on routers R2, R3, and R4 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv4 addresses.

```
[edit interfaces]
user@R1# set ge-0/0/0 unit 0 description R1->R3
user@R1# set ge-0/0/0 unit 0 family inet address 10.0.0.1/30
user@R1# set ge-0/0/1 unit 0 description R1->R2
```

```
user@R1# set ge-0/0/1 unit 0 family inet address 10.0.1.2/30
user@R1# set ge-0/0/2 unit 0 description R1->R4
user@R1# set ge-0/0/2 unit 0 family inet address 10.0.4.1/30
```

2. Configure the loopback address.

```
[edit interfaces]
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

3. Configure the autonomous system for BGP hosts.

```
[edit routing-options]
user@R1# set autonomous-system 65000
```

4. Configure EBGP on the external edge routers.

```
[edit protocols]
user@R1# set bgp group external type external
user@R1# set bgp group external import bw-dis
user@R1# set bgp group external peer-as 65001
user@R1# set bgp group external multipath
user@R1# set bgp group external neighbor 10.0.1.1
user@R1# set bgp group external neighbor 10.0.0.2
user@R1# set bgp group external2 type external
user@R1# set bgp group external2 peer-as 65002
```

5. Define a bandwidth distribution policy to assign a high bandwidth community to traffic destined to Router R3.

```
[edit policy-options]
user@R1# set policy-statement bw-dis term a from protocol bgp
user@R1# set policy-statement bw-dis term a from neighbor 10.0.1.1
user@R1# set policy-statement bw-dis term a then community add bw-high
user@R1# set policy-statement bw-dis term a then accept
```

6. Define a bandwidth distribution policy to assign a low bandwidth community to traffic destined to Router R2.

```
[edit policy-options]
user@R1# set policy-statement bw-dis term b from protocol bgp
user@R1# set policy-statement bw-dis term b from neighbor 10.0.0.2
user@R1# set policy-statement bw-dis term b then community add bw-low
user@R1# set policy-statement bw-dis term b then accept
```

7. Enable the feature to advertise aggregated bandwidth of 80,000,000 bytes to EBGP peer Router R4 over BGP sessions.

```
[edit policy-options]
user@R1# set policy-statement aggregate_bw_and_limit_capacity then aggregate-bandwidth
user@R1# set policy-statement aggregate_bw_and_limit_capacity then limit-bandwidth 80000000
user@R1# set policy-statement aggregate_bw_and_limit_capacity then accept
```

8. Apply the aggregate_bw_and limit_capacity policy to EBGP group `external2`.

```
[edit protocols]
user@R1# set bgp group external2 neighbor 10.0.4.2 export aggregate_bw_and_limit_capacity
```

9. Define a load balancing policy.

```
[edit policy-options]
user@R1# set policy-statement loadbal from route-filter 10.0.0.0/16 orlonger
user@R1# set policy-statement loadbal then load-balance per-packet
```

10. Apply the load balancing policy.

```
[edit routing-options]
user@R1# set forwarding-table export loadbal
```

11. Configure the BGP community members. The first 16-bit number represents the local autonomous system. The second 32-bit number represents the link bandwidth in bytes per second. Configure a `bw-high` community with 60 percent of a 1-Gbps link and another community `bw-low` with 40 percent of a 1-Gbps link.

Configure 60 percent of a 1-Gbps link to bw-high community and 40 percent to bw-low community.

```
[edit policy-options]
user@R1# set community bw-high members bandwidth:65000:60000000
user@R1# set community bw-low members bandwidth:65000:40000000
```

Results

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

```
[edit]
user@R1# show interfaces
interfaces {
    ge-0/0/0 {
        unit 0 {
            description R1->R3;
            family inet {
                address 10.0.0.1/30;
            }
        }
    }
    ge-0/0/1 {
        unit 0 {
            description R1->R2;
            family inet {
                address 10.0.1.2/30;
            }
        }
    }
    ge-0/0/2 {
        unit 0 {
            description R1->R4;
            family inet {
                address 10.0.4.1/30;
            }
        }
    }
    lo0 {
```

```
        unit 0 {
            family inet {
                address 192.168.0.1/32;
            }
        }
    }
}
```

```
[edit]
user@R1# show protocols
protocols {
    bgp {
        group external {
            type external;
            import bw-dis;
            peer-as 65001;
            multipath;
            neighbor 10.0.1.1;
            neighbor 10.0.0.2;
        }
        group external2 {
            type external;
            peer-as 65002;
            neighbor 10.0.4.2 {
                export aggregate_bw_and_limit_capacity;
            }
        }
    }
}
```

```
[edit]
user@R1# show routing-options
routing-options {
    autonomous-system 65000;
    forwarding-table {
        export loadbal;
```

```
    }
}
```

```
[edit]
user@R1# show policy-options
policy-options {
    policy-statement bw-dis {
        term a {
            from {
                protocol bgp;
                neighbor 10.0.1.1;
            }
            then {
                community add bw-high;
                accept;
            }
        }
        term b {
            from {
                protocol bgp;
                neighbor 10.0.0.2;
            }
            then {
                community add bw-low;
                accept;
            }
        }
    }
    policy-statement aggregate_bw_and_limit_capacity {
        then {
            aggregate-bandwidth;
            limit-bandwidth 80000000;
            accept;
        }
    }
    policy-statement loadbal {
        from {
            route-filter 10.0.0.0/16 orlonger;
        }
        then {
            load-balance per-packet;
```

```
        }
    }
    community bw-high members bandwidth:65000:60000000;
    community bw-low members bandwidth:65000:40000000;
}
```

Verification

IN THIS SECTION

- Verifying BGP Session Is Established | **607**
- Verifying That the Aggregate Bandwidth Is Present in Each Path | **608**
- Verifying That Router R1 Is Advertising the Aggregate Bandwidth to Its Neighbor Router R4 | **609**

Verifying BGP Session Is Established

Purpose

To verify that BGP peering is complete and a BGP session is established between the routers,

Action

```
user@R1> show bgp summary
Groups: 2 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                     12          8          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.0.0.2              65001        153        149       0       0     1:07:23 4/6/6/0
0/0/0/0
10.0.1.1              65001        229        226       0       0     1:41:44 4/6/6/0
0/0/0/0
10.0.4.2              65002       1227       1227       0       0     9:10:27 0/0/0/0
0/0/0/0
```

Meaning

Router R1 has completed peering with Routers R2, R3, and R4.

Verifying That the Aggregate Bandwidth Is Present in Each Path

Purpose

To verify that the extended community is present for each route path.

Action

From operational mode, run the `show route protocol bgp detail` command.

```
user@R1> show route 10.0.2.0 protocol bgp detail
inet.0: 20 destinations, 26 routes (20 active, 0 holddown, 0 hidden)
10.0.2.0/30 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 0
                Address: 0xb618990
                Next-hop reference count: 3
                Source: 10.0.1.1
                Next hop: 10.0.0.2 via ge-0/0/0.0 balance 40%
                Session Id: 0x0
                Next hop: 10.0.1.1 via ge-0/0/1.0 balance 60%, selected
                Session Id: 0x0
                State: <Active Ext>
                Local AS: 65000 Peer AS: 65001
                Age: 20:33
                Validation State: unverified
                Task: BGP_65001.10.0.1.1
                Announcement bits (3): 0-KRT 2-BGP_Listen.0.0.0.0+179 3-BGP_RT_Background
                AS path: 65001 I
                Communities: bandwidth:65000:60000000
                Accepted Multipath
                Localpref: 100
                Router ID: 128.49.121.137
         BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 595
                Address: 0xb7a1330
                Next-hop reference count: 9
                Source: 10.0.0.2
```

```
                Next hop: 10.0.0.2 via ge-0/0/0.0, selected
                Session Id: 0x141
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Active preferred
                Local AS: 65000 Peer AS: 65001
                Age: 20:33
                Validation State: unverified
                Task: BGP_65001.10.0.0.2
                AS path: 65001 I
                Communities: bandwidth:65000:40000000
                Accepted MultipathContrib
                Localpref: 100
                Router ID: 128.49.121.132
```

Meaning

Verifying That Router R1 Is Advertising the Aggregate Bandwidth to Its Neighbor Router R4

Purpose

To verify that Router R1 is advertising the aggregate bandwidth to its external neighbors.

Action

```
user@R1> show route advertising-protocol bgp 10.0.4.2 10.0.2.0/30 detail
inet.0: 20 destinations, 26 routes (20 active, 0 holddown, 0 hidden)
* 10.0.2.0/30 (2 entries, 1 announced)
 BGP group external2 type External
     Nexthop: Self
     AS path: [65000] 65001 I
     Communities: bandwidth:65000:80000000
```

Meaning

Router R1 is advertising the aggregated bandwidth of 80,000,000 bytes to its neighbors.

SEE ALSO

policy-statement

Understanding the Advertisement of Multiple Paths to a Single Destination in BGP

BGP peers advertise routes to each other in update messages. BGP stores its routes in the Junos OS routing table (`inet.0`). For each prefix in the routing table, the routing protocol process selects a single best path, called the active path. Unless you configure BGP to advertise multiple paths to the same destination, BGP advertises only the active path.

Instead of advertising only the active path to a destination, you can configure BGP to advertise multiple paths to the destination. Within an autonomous system (AS), the availability of multiple exit points to reach a destination provides the following benefits:

- Fault tolerance—Path diversity leads to reduction in restoration time after failure. For instance, a border after receiving multiple paths to the same destination can precompute a backup path and have it ready so that when the primary path becomes invalid, the border routing device can use the backup to quickly restore connectivity. Without a backup path, the restoration time depends on BGP reconvergence, which includes withdraw and advertisement messages in the network before a new best path can be learned.
- Load balancing—The availability of multiple paths to reach the same destination enables load balancing of traffic, if the routing within the AS meets certain constraints.
- Maintenance—The availability of alternate exit points allows for graceful maintenance operation of routers.

The following limitations apply to advertising multiple routes in BGP:

- Address families supported:
  - IPv4 unicast (`family inet unicast`)
  - IPv6 unicast (`family inet6 unicast`)
  - IPv4 labeled unicast (`family inet labeled-unicast`)
  - IPv6 labeled unicast (`family inet6 labeled-unicast`)
  - IPv4 VPN unicast (`family inet-vpn unicast`)
  - IPv6 VPN unicast (`family inet6-vpn unicast`)

The following example shows the configuration of IPv4 VPN unicast and IPv6 VPN unicast families:

```
bgp {
    group <group-name> {
        family inet-vpn unicast {
            add-path {
                send {
                    include-backup-path include-backup-path;
                    multipath;
                    path-count path-count;
                    path-selection-mode {
                        (all-paths | equal-cost-paths);
                    }
                    prefix-policy [ policy-names ... ];
                }
            receive;
        }
        family inet6-vpn unicast {
            add-path {
                send {
                    include-backup-path include-backup-path;
                    multipath;
                    path-count path-count;
                    path-selection-mode {
                        (all-paths | equal-cost-paths);
                    }
                    prefix-policy [ policy-names ... ];
                }
            receive;
        }
    }
}
```

- We support `add-path` for internal BGP (IBGP) and external BGP (EBGP) peers.

NOTE:

- We support add-path **receive** for IBGP and EBGP peers.
- We support add-path **send only** for IBGP peers.

- We do not support add-path **send** for EBGP peers. When you try to commit the configuration for add-path **send** for EBGP peers, the CLI throws a commit error.

- Master instance only. No support for routing instances.
- Graceful restart and nonstop active routing (NSR) are supported.
- No BGP Monitoring Protocol (BMP) support.
- Prefix policies enable you to filter routes on a router that is configured to advertise multiple paths to a destination. Prefix policies can only match prefixes. They cannot match route attributes, and they cannot change the attributes of routes.

Starting in Junos OS Release 18.4R1, BGP can advertise a maximum of 2 add-path routes in addition to the multiple ECMP paths.

To advertise all add-paths up to 64 add-paths or only equal-cost-paths, include `path-selection-mode` at the `[edit protocols bgp group group-name family name addpath send]` hierarchy level. You cannot enable both `multipath` and `path-selection-mode` at the same time.

SEE ALSO

Understanding BGP Path Selection | 12

Example: Advertising Multiple Paths in BGP

IN THIS SECTION

- Requirements | 613
- Overview | 613
- Configuration | 615
- Verification | 642

In this example, BGP routers are configured to advertise multiple paths instead of advertising only the active path. Advertising multiple paths in BGP is specified in RFC 7911, *Advertisement of Multiple Paths in BGP*.

Requirements

This example uses the following hardware and software components:

- Eight BGP-enabled devices.
- Five of the BGP-enabled devices do not necessarily need to be routers. For example, they can be EX Series Ethernet Switches.
- Three of the BGP-enabled devices are configured to send multiple paths or receive multiple paths (or both send and receive multiple paths). These three BGP-enabled devices must be M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers.
- The three routers must be running Junos OS Release 11.4 or later.

Overview

IN THIS SECTION

- Topology Diagram | 614

The following statements are used for configuring multiple paths to a destination:

```
[edit protocols bgp group group-name family family]
add-path {
    receive;
    send {
        include-backup-path include-backup-path;
        multipath;
        path-count path-count;
        path-selection-mode {
            (all-paths | equal-cost-paths);
        }
        prefix-policy [ policy-names ... ];
    }
}
```

In this example, Router R5, Router R6, and Router R7 redistribute static routes into BGP. Router R1 and Router R4 are route reflectors. Router R2 and Router R3 are clients to Route Reflector R1. Router R8 is a client to Route Reflector R4.

Route reflection is optional when multiple-path advertisement is enabled in BGP.

With the `add-path send path-count 6` configuration, Router R1 is configured to send up to six paths (per destination) to Router R4.

With the `add-path receive` configuration, Router R4 is configured to receive multiple paths from Router R1.

With the `add-path send path-count 6` configuration, Router R4 is configured to send up to six paths to Router R8.

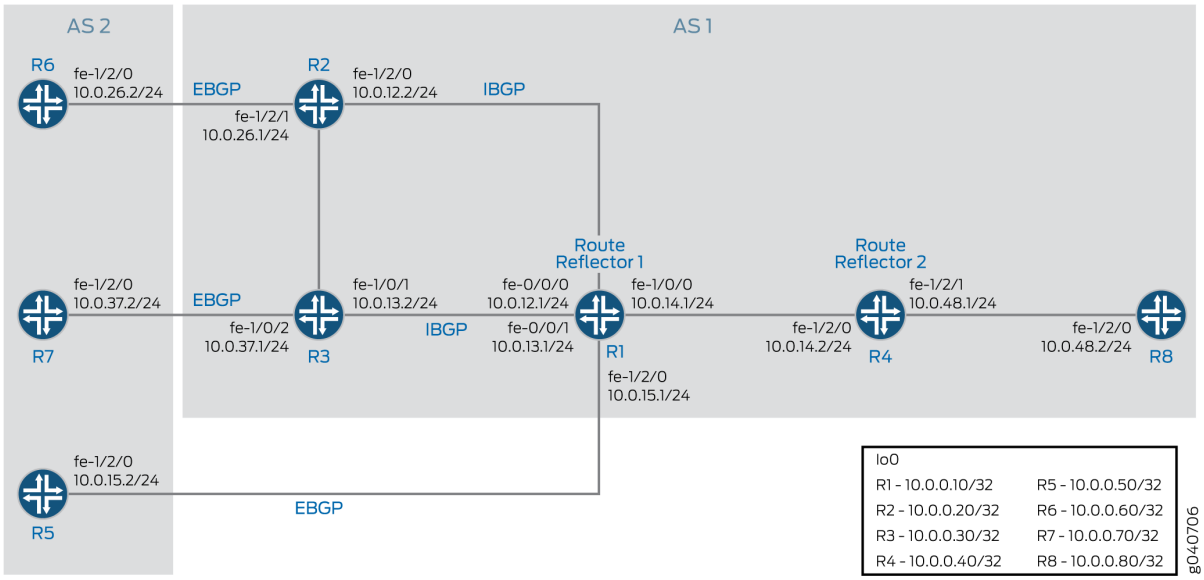
With the `add-path receive` configuration, Router R8 is configured to receive multiple paths from Router R4.

The `add-path send prefix-policy allow_199` policy configuration (along with the corresponding route filter) limits Router R4 to sending multiple paths for only the 172.16.199.1/32 route.

Topology Diagram

Figure 47 on page 614 shows the topology used in this example.

Figure 47: Advertisement of Multiple Paths in BGP



Configuration

IN THIS SECTION

- CLI Quick Configuration | 615
- Configuring Router R1 | 619
- Configuring Router R2 | 622
- Configuring Router R3 | 625
- Configuring Router R4 | 628
- Configuring Router R5 | 632
- Configuring Router R6 | 635
- Configuring Router R7 | 637
- Configuring Router R8 | 639
- Results | 641

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 R1

```
set interfaces fe-0/0/0 unit 12 family inet address 10.0.12.1/24
set interfaces fe-0/0/1 unit 13 family inet address 10.0.13.1/24
set interfaces fe-1/0/0 unit 14 family inet address 10.0.14.1/24
set interfaces fe-1/2/0 unit 15 family inet address 10.0.15.1/24
set interfaces lo0 unit 10 family inet address 10.0.0.10/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.10
set protocols bgp group rr cluster 10.0.0.10
set protocols bgp group rr neighbor 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.30
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.2 local-address 10.0.15.1
set protocols bgp group e1 neighbor 10.0.15.2 peer-as 2
set protocols bgp group rr_rr type internal
```

```
set protocols bgp group rr_rr local-address 10.0.0.10
set protocols bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send path-count 6
set protocols ospf area 0.0.0.0 interface lo0.10 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.12
set protocols ospf area 0.0.0.0 interface fe-0/0/1.13
set protocols ospf area 0.0.0.0 interface fe-1/0/0.14
set protocols ospf area 0.0.0.0 interface fe-1/2/0.15
set routing-options router-id 10.0.0.10
set routing-options autonomous-system 1
```

Router R2

```
set interfaces fe-1/2/0 unit 21 family inet address 10.0.12.2/24
set interfaces fe-1/2/1 unit 26 family inet address 10.0.26.1/24
set interfaces lo0 unit 20 family inet address 10.0.0.20/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.2 peer-as 2
set protocols ospf area 0.0.0.0 interface lo0.20 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.21
set protocols ospf area 0.0.0.0 interface fe-1/2/1.28
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 1
```

Router R3

```
set interfaces fe-1/0/1 unit 31 family inet address 10.0.13.2/24
set interfaces fe-1/0/2 unit 37 family inet address 10.0.37.1/24
set interfaces lo0 unit 30 family inet address 10.0.0.30/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.30
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.2 peer-as 2
set protocols ospf area 0.0.0.0 interface lo0.30 passive
set protocols ospf area 0.0.0.0 interface fe-1/0/1.31
set protocols ospf area 0.0.0.0 interface fe-1/0/2.37
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 1
```

Router R4

```
set interfaces fe-1/2/0 unit 41 family inet address 10.0.14.2/24
set interfaces fe-1/2/1 unit 48 family inet address 10.0.48.1/24
set interfaces lo0 unit 40 family inet address 10.0.0.40/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.40
set protocols bgp group rr family inet unicast add-path receive
set protocols bgp group rr neighbor 10.0.0.10
set protocols bgp group rr_client type internal
set protocols bgp group rr_client local-address 10.0.0.40
set protocols bgp group rr_client cluster 10.0.0.40
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send path-
count 6
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send prefix-
policy allow_199
set protocols ospf area 0.0.0.0 interface fe-1/2/0.41
set protocols ospf area 0.0.0.0 interface lo0.40 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/1.48
set policy-options policy-statement allow_199 from route-filter 172.16.199.1/32 exact
set policy-options policy-statement allow_199 term match_199 from prefix-list match_199
set policy-options policy-statement allow_199 then add-path send-count 20
set policy-options policy-statement allow_199 then accept
set routing-options autonomous-system 1
```

Router R5

```
set interfaces fe-1/2/0 unit 51 family inet address 10.0.15.2/24
set interfaces lo0 unit 50 family inet address 10.0.0.50/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.1 export s2b
set protocols bgp group e1 neighbor 10.0.15.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then as-path-expand 2
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
set routing-options static route 172.16.198.1/32 reject
```

Router R6

```
set interfaces fe-1/2/0 unit 62 family inet address 10.0.26.2/24
set interfaces lo0 unit 60 family inet address 10.0.0.60/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.1 export s2b
set protocols bgp group e1 neighbor 10.0.26.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
set routing-options static route 172.16.198.1/32 reject
```

Router R7

```
set interfaces fe-1/2/0 unit 73 family inet address 10.0.37.2/24
set interfaces lo0 unit 70 family inet address 10.0.0.70/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.1 export s2b
set protocols bgp group e1 neighbor 10.0.37.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
```

Router R8

```
set interfaces fe-1/2/0 unit 84 family inet address 10.0.48.2/24
set interfaces lo0 unit 80 family inet address 10.0.0.80/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.80
set protocols bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
set protocols ospf area 0.0.0.0 interface lo0.80 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.84
set routing-options autonomous-system 1
```

Configuring Router R1

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 Junos OS CLI User Guide.

To configure Router R1:

1. Configure the interfaces to Router R2, Router R3, Router R4, and Router R5, and configure the loopback (lo0) interface.

```
[edit interfaces]
user@R1# set fe-0/0/0 unit 12 family inet address 10.0.12.1/24
user@R1# set fe-0/0/1 unit 13 family inet address 10.0.13.1/24
user@R1# set fe-1/0/0 unit 14 family inet address 10.0.14.1/24
user@R1# set fe-1/2/0 unit 15 family inet address 10.0.15.1/24
user@R1#set lo0 unit 10 family inet address 10.0.0.10/32
```

2. Configure BGP on the interfaces, and configure IBGP route reflection.

```
[edit protocols bgp]
user@R1# set group rr type internal
user@R1# set group rr local-address 10.0.0.10
user@R1# set group rr cluster 10.0.0.10
user@R1# set group rr neighbor 10.0.0.20
user@R1# set group rr neighbor 10.0.0.30
user@R1# set group rr_rr type internal
user@R1# set group rr_rr local-address 10.0.0.10
user@R1# set group e1 type external
user@R1# set group e1 neighbor 10.0.15.2 local-address 10.0.15.1
user@R1# set group e1 neighbor 10.0.15.2 peer-as 2
```

3. Configure Router R1 to send up to six paths to its neighbor, Router R4.

   The destination of the paths can be any destination that Router R1 can reach through multiple paths.

```
[edit protocols bgp]
user@R1# set group rr_rr neighbor 10.0.0.40 family inet unicast add-path send path-count 6
```

4. Configure OSPF on the interfaces.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface lo0.10 passive
user@R1# set area 0.0.0.0 interface fe-0/0/0.12
user@R1# set area 0.0.0.0 interface fe-0/0/1.13
user@R1# set area 0.0.0.0 interface fe-1/0/0.14
user@R1# set area 0.0.0.0 interface fe-1/2/0.15
```

5. Configure the router ID and the autonomous system number.

```
[edit routing-options]
user@R1# set router-id 10.0.0.10
user@R1# set autonomous-system 1
```

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

```
user@R1# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
fe-0/0/0 {
    unit 12 {
        family inet {
            address 10.0.12.1/24;
        }
    }
}
fe-0/0/1 {
    unit 13 {
        family inet {
            address 10.0.13.1/24;
        }
```

```
    }
}
fe-1/0/0 {
    unit 14 {
        family inet {
            address 10.0.14.1/24;
        }
    }
}
fe-1/2/0 {
    unit 15 {
        family inet {
            address 10.0.15.1/24;
        }
    }
}
lo0 {
    unit 10 {
        family inet {
            address 10.0.0.10/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.10;
        cluster 10.0.0.10;
        neighbor 10.0.0.20;
        neighbor 10.0.0.30;
    }
    group e1 {
        type external;
        neighbor 10.0.15.2 {
            local-address 10.0.15.1;
            peer-as 2;
        }
    }
    group rr_rr {
```

```
        type internal;
        local-address 10.0.0.10;
        neighbor 10.0.0.40 {
            family inet {
                unicast {
                    add-path {
                        send {
                            path-count 6;
                        }
                    }
                }
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.10 {
            passive;
        }
        interface fe-0/0/0.12;
        interface fe-0/0/1.13;
        interface fe-1/0/0.14;
        interface fe-1/2/0.15;
    }
}
```

```
user@R1# show routing-options
router-id 10.0.0.10;
 autonomous-system 1;
```

Configuring Router R2

Step-by-Step Procedure

To configure Router R2:

1. Configure the loopback (lo0) interface and the interfaces to Router R6 and Router R1.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 21 family inet address 10.0.12.2/24
user@R2# set fe-1/2/1 unit 26 family inet address 10.0.26.1/24
user@R2# set lo0 unit 20 family inet address 10.0.0.20/32
```

2. Configure BGP and OSPF on Router R2's interfaces.

```
[edit protocols]
user@R2# set bgp group rr type internal
user@R2# set bgp group rr local-address 10.0.0.20
user@R2# set bgp group e1 type external
user@R2# set bgp group e1 neighbor 10.0.26.2 peer-as 2
user@R2# set ospf area 0.0.0.0 interface lo0.20 passive
user@R2# set ospf area 0.0.0.0 interface fe-1/2/0.21
user@R2# set ospf area 0.0.0.0 interface fe-1/2/1.28
```

3. For routes sent from Router R2 to Router R1, advertise Router R2 as the next hop, because Router R1 does not have a route to Router R6's address on the 10.0.26.0/24 network.

```
[edit]
user@R2# set policy-options policy-statement set_nh_self then next-hop self
user@R2# set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
```

4. Configure the autonomous system number.

```
[edit]
user@R2# set routing-options autonomous-system 1
```

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

```
user@R2# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`,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@R2# show interfaces
fe-1/2/0 {
    unit 21 {
        family inet {
            address 10.0.12.2/24;
        }
    }
}
fe-1/2/1 {
    unit 26 {
        family inet {
            address 10.0.26.1/24;
        }
    }
}
lo0 {
    unit 20 {
        family inet {
            address 10.0.0.20/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.20;
        neighbor 10.0.0.10 {
            export set_nh_self;
        }
    }
    group e1 {
        type external;
        neighbor 10.0.26.2 {
```

```
            peer-as 2;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.20 {
            passive;
        }
        interface fe-1/2/0.21;
        interface fe-1/2/1.28;
    }
}
```

```
user@R2# show policy-options
policy-statement set_nh_self {
    then {
        next-hop self;
    }
}
```

```
user@R2# show routing-options
 autonomous-system 1;
```

Configuring Router R3

Step-by-Step Procedure

To configure Router R3:

1. Configure the loopback (lo0) interface and the interfaces to Router R7 and Router R1.

```
[edit interfaces]
user@R3# set fe-1/0/1 unit 31 family inet address 10.0.13.2/24
user@R3# set fe-1/0/2 unit 37 family inet address 10.0.37.1/24
user@R3# set lo0 unit 30 family inet address 10.0.0.30/32
```

2. Configure BGP and OSPF on Router R3's interfaces.

```
[edit protocols]
user@R3# set bgp group rr type internal
user@R3# set bgp group rr local-address 10.0.0.30
user@R3# set bgp group e1 type external
user@R3# set bgp group e1 neighbor 10.0.37.2 peer-as 2
user@R3# set ospf area 0.0.0.0 interface lo0.30 passive
user@R3# set ospf area 0.0.0.0 interface fe-1/0/1.31
user@R3# set ospf area 0.0.0.0 interface fe-1/0/2.37
```

3. For routes sent from Router R3 to Router R1, advertise Router R3 as the next hop, because Router R1 does not have a route to Router R7's address on the 10.0.37.0/24 network.

```
[edit]
user@R3# set policy-options policy-statement set_nh_self then next-hop self
user@R3# set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
```

4. Configure the autonomous system number.

```
[edit]
user@R3# set routing-options autonomous-system 1
```

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

```
user@R3# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R3# show interfaces
fe-1/0/1 {
    unit 31 {
        family inet {
```

```
            address 10.0.13.2/24;
        }
    }
}
fe-1/0/2 {
    unit 37 {
        family inet {
            address 10.0.37.1/24;
        }
    }
}
lo0 {
    unit 30 {
        family inet {
            address 10.0.0.30/32;
        }
    }
}
```

```
user@R3# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.30;
        neighbor 10.0.0.10 {
            export set_nh_self;
        }
    }
    group e1 {
        type external;
        neighbor 10.0.37.2 {
            peer-as 2;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.30 {
            passive;
        }
        interface fe-1/0/1.31;
```

```
        interface fe-1/0/2.37;
    }
}
user@R3# show policy-options
policy-statement set_nh_self {
    then {
        next-hop self;
    }
}
```

```
user@R3# show routing-options
 autonomous-system 1;
```

Configuring Router R4

Step-by-Step Procedure

To configure Router R4:

1. Configure the interfaces to Router R1 and Router R8, and configure the loopback (lo0) interface.

```
[edit interfaces]
user@R4# set fe-1/2/0 unit 41 family inet address 10.0.14.2/24
user@R4# set fe-1/2/1 unit 48 family inet address 10.0.48.1/24
user@R4# set lo0 unit 40 family inet address 10.0.0.40/32
```

2. Configure BGP on the interfaces, and configure IBGP route reflection.

```
[edit protocols bgp]
user@R4# set group rr type internal
user@R4# set group rr local-address 10.0.0.40
user@R4# set group rr neighbor 10.0.0.10
user@R4# set group rr_client type internal
user@R4# set group rr_client local-address 10.0.0.40
user@R4# set group rr_client cluster 10.0.0.40
```

3. Configure Router R4 to send up to six paths to its neighbor, Router R8.

The destination of the paths can be any destination that Router R4 can reach through multiple paths.

```
[edit protocols bgp]
user@R4# set group rr_client neighbor 10.0.0.80 family inet unicast add-path send path-count 6
```

4. Configure Router R4 to receive multiple paths from its neighbor, Router R1.

   The destination of the paths can be any destination that Router R1 can reach through multiple paths.

```
[edit protocols bgp group rr family inet unicast]
user@R4# set add-path receive
```

5. Configure OSPF on the interfaces.

```
[edit protocols ospf area 0.0.0.0]
user@R4# set interface fe-1/2/0.41
user@R4# set interface lo0.40 passive
user@R4# set interface fe-1/2/1.48
```

6. Configure a policy that allows Router R4 to send Router R8 multiple paths to the 172.16.199.1/32 route.

   - Router R4 receives multiple paths for the 172.16.198.1/32 route and the 172.16.199.1/32 route. However, because of this policy, Router R4 only sends multiple paths for the 172.16.199.1/32 route.

```
[edit protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast]
user@R4# set add-path send prefix-policy allow_199
[edit policy-options policy-statement allow_199]
user@R4# set from route-filter 172.16.199.1/32 exact
user@R4# set then accept
```

   - Router R4 can also be configured to send up-to 20 BGP `add-path` routes for a subset of *add-path advertised prefixes*.

```
[edit policy-options policy-statement allow_199]
user@R4# set term match_199 from prefix-list match_199
user@R4# set then add-path send-count 20
```

7. Configure the autonomous system number.

```
[edit routing-options]
user@R4# set autonomous-system 1
```

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

```
user@R4# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R4# show interfaces
fe-1/2/0 {
    unit 41 {
        family inet {
            address 10.0.14.2/24;
        }
    }
}
fe-1/2/1 {
    unit 48 {
        family inet {
            address 10.0.48.1/24;
        }
    }
}
lo0 {
    unit 40 {
        family inet {
            address 10.0.0.40/32;
        }
```

```
    }
}
```

```
user@R4# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.40;
        family inet {
            unicast {
                add-path {
                    receive;
                }
            }
        }
        neighbor 10.0.0.10;
    }
    group rr_client {
        type internal;
        local-address 10.0.0.40;
        cluster 10.0.0.40;
        neighbor 10.0.0.80 {
            family inet {
                unicast {
                    add-path {
                        send {
                            path-count 6;
                            prefix-policy allow_199;
                        }
                    }
                }
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.40 {
            passive;
        }
        interface fe-1/2/0.41;
```

```
        interface fe-1/2/1.48;
    }
}
```

```
user@R4# show policy-options
policy-statement allow_199 {
    from {
        route-filter 172.16.199.1/32 exact;
    }
    from term match_199 {
        prefix-list match_199;
    }
        then add-path send-count 20;
    then accept;
}
```

```
user@R4# show routing-options
 autonomous-system 1;
```

Configuring Router R5

Step-by-Step Procedure

To configure Router R5:

1. Configure the loopback (lo0) interface and the interface to Router R1.

```
[edit interfaces]
user@R5# set fe-1/2/0 unit 51 family inet address 10.0.15.2/24
user@R5# set lo0 unit 50 family inet address 10.0.0.50/32
```

2. Configure BGP on Router R5's interface.

```
[edit protocols bgp group e1]
user@R5# set type external
user@R5# set neighbor 10.0.15.1 peer-as 1
```

3. Create static routes for redistribution into BGP.

```
[edit routing-options]
user@R5# set static route 172.16.199.1/32 reject
user@R5# set static route 172.16.198.1/32 reject
```

4. Redistribute static and direct routes into BGP.

```
[edit protocols bgp group e1 neighbor 10.0.15.1]
user@R5# set export s2b
[edit policy-options policy-statement s2b]
user@R5# set from protocol static
user@R5# set from protocol direct
user@R5# set then as-path-expand 2
user@R5# set then accept
```

5. Configure the autonomous system number.

```
[edit routing-options]
user@R5# set autonomous-system 2
```

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

```
user@R5# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R5# show interfaces
fe-1/2/0 {
    unit 51 {

        family inet {
            address 10.0.15.2/24;
```

```
        }
    }
}
lo0 {
    unit 50 {
        family inet {
            address 10.0.0.50/32;
        }
    }
}
```

```
user@R5# show protocols
bgp {
    group e1 {
        type external;
        neighbor 10.0.15.1 {
            export s2b;
            peer-as 1;
        }
    }
}
```

```
user@R5# show policy-options
policy-statement s2b {
    from protocol [ static direct ];
    then {
        as-path-expand 2;
        accept;
    }
}
```

```
user@R5# show routing-options
static {
    route 172.16.198.1/32 reject;
    route 172.16.199.1/32 reject;
}
 autonomous-system 2;
```

Configuring Router R6

Step-by-Step Procedure

To configure Router R6:

1. Configure the loopback (lo0) interface and the interface to Router R2.

```
[edit interfaces]
user@R6# set fe-1/2/0 unit 62 family inet address 10.0.26.2/24
user@R6# set lo0 unit 60 family inet address 10.0.0.60/32
```

2. Configure BGP on Router R6's interface.

```
[edit protocols]
user@R6# set bgp group e1 type external
user@R6# set bgp group e1 neighbor 10.0.26.1 peer-as 1
```

3. Create static routes for redistribution into BGP.

```
[edit]
user@R6# set routing-options static route 172.16.199.1/32 reject
user@R6# set routing-options static route 172.16.198.1/32 reject
```

4. Redistribute static and direct routes from Router R6's routing table into BGP.

```
[edit protocols bgp group e1 neighbor 10.0.26.1]
user@R6# set export s2b
[edit policy-options policy-statement s2b]
user@R6# set from protocol static
user@R6# set from protocol direct
user@R6# set then accept
```

5. Configure the autonomous system number.

```
[edit routing-options]
user@R6# set autonomous-system 2
```

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

```
user@R6# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R6# show interfaces
fe-1/2/0 {
    unit 62 {

        family inet {
            address 10.0.26.2/24;
        }
    }
}
lo0 {
    unit 60 {
        family inet {
            address 10.0.0.60/32;
        }
    }
}
```

```
user@R6# show protocols
bgp {
    group e1 {
        type external;
        neighbor 10.0.26.1 {
            export s2b;
            peer-as 1;
        }
```

```
    }
}
```

```
user@R6# show policy-options
policy-statement s2b {
    from protocol [ static direct ];
    then accept;
}
```

```
user@R6# show routing-options
static {
    route 172.16.198.1/32 reject;
    route 172.16.199.1/32 reject;
}
 autonomous-system 2;
```

Configuring Router R7

Step-by-Step Procedure

To configure Router R7:

1. Configure the loopback (lo0) interface and the interface to Router R3.

```
[edit interfaces]
user@R7# set fe-1/2/0 unit 73 family inet address 10.0.37.2/24
user@R7# set lo0 unit 70 family inet address 10.0.0.70/32
```

2. Configure BGP on Router R7's interface.

```
[edit protocols bgp group e1]
user@R7# set type external
user@R7# set neighbor 10.0.37.1 peer-as 1
```

3. Create a static route for redistribution into BGP.

```
[edit]
user@R7# set routing-options static route 172.16.199.1/32 reject
```

4. Redistribute static and direct routes from Router R7's routing table into BGP.

```
[edit protocols bgp group e1 neighbor 10.0.37.1]
user@R7# set export s2b
[edit policy-options policy-statement s2b]
user@R7# set from protocol static
user@R7# set from protocol direct
user@R7# set then accept
```

5. Configure the autonomous system number.

```
[edit routing-options]
user@R7# set autonomous-system 2
```

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

```
user@R7# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R7# show interfaces
fe-1/2/0 {
    unit 73 {

        family inet {
            address 10.0.37.2/24;
        }
    }
```

```
}
lo0 {
    unit 70 {
        family inet {
            address 10.0.0.70/32;
        }
    }
}
```

```
user@R7# show protocols
bgp {
    group e1 {
        type external;
        neighbor 10.0.37.1 {
            export s2b;
            peer-as 1;
        }
    }
}
```

```
user@R7# show policy-options
policy-statement s2b {
    from protocol [ static direct ];
    then accept;
}
```

```
user@R7# show routing-options
static {
    route 172.16.199.1/32 reject;
}
 autonomous-system 2;
```

Configuring Router R8

Step-by-Step Procedure

To configure Router R8:

1. Configure the loopback (lo0) interface and the interface to Router R4.

```
[edit interfaces]
user@R8# set fe-1/2/0 unit 84 family inet address 10.0.48.2/24
user@R8# set lo0 unit 80 family inet address 10.0.0.80/32
```

2. Configure BGP and OSPF on Router R8's interface.

```
[edit protocols]
user@R8# set bgp group rr type internal
user@R8# set bgp group rr local-address 10.0.0.80
user@R8# set ospf area 0.0.0.0 interface lo0.80 passive
user@R8# set ospf area 0.0.0.0 interface fe-1/2/0.84
```

3. Configure Router R8 to receive multiple paths from its neighbor, Router R4.

   The destination of the paths can be any destination that Router R4 can reach through multiple paths.

```
[edit protocols]
user@R8# set bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
```

4. Configure the autonomous system number.

```
[edit]
user@R8# set routing-options autonomous-system 1
```

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

```
user@R8# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R8# show interfaces
fe-1/2/0 {

    unit 84 {

        family inet {
            address 10.0.48.2/24;
        }
    }
}
lo0 {
    unit 80 {
        family inet {
            address 10.0.0.80/32;
        }
    }
}
```

```
user@R8# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.80;
        neighbor 10.0.0.40 {
            family inet {
                unicast {
                    add-path {
                        receive;
                    }
                }
            }
        }
    }
}
ospf {
```

```
    area 0.0.0.0 {
        interface lo0.80 {
            passive;
        }
        interface fe-1/2/0.84;
    }
}
```

```
user@R8# show routing-options
 autonomous-system 1;
```

Verification

IN THIS SECTION

- Verifying That the BGP Peers Have the Ability to Send and Receive Multiple Paths | **642**
- Verifying That Router R1 Is Advertising Multiple Paths | **643**
- Verifying That Router R4 Is Receiving and Advertising Multiple Paths | **644**
- Verifying That Router R8 Is Receiving Multiple Paths | **645**
- Checking the Path ID | **646**

Confirm that the configuration is working properly.

Verifying That the BGP Peers Have the Ability to Send and Receive Multiple Paths

Purpose

Make sure that one or both of the following strings appear in the output of the `show bgp neighbor` command:

- `NLRI's for which peer can receive multiple paths: inet-unicast`
- `NLRI's for which peer can send multiple paths: inet-unicast`

Action

```
user@R1> show bgp neighbor 10.0.0.40
Peer: 10.0.0.40+179 AS 1       Local: 10.0.0.10+64227 AS 1
  Type: Internal    State: Established    Flags: <Sync>
...  NLRI's for which peer can receive multiple paths: inet-unicast
...
```

```
user@R4> show bgp neighbor 10.0.0.10
Peer: 10.0.0.10+64227 AS 1     Local: 10.0.0.40+179 AS 1
  Type: Internal    State: Established    Flags: <Sync>
...
  NLRI's for which peer can send multiple paths: inet-unicast
...
```

```
user@R4> show bgp neighbor 10.0.0.80
Peer: 10.0.0.80+55416 AS 1     Local: 10.0.0.40+179 AS 1
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  ,,,
  NLRI's for which peer can receive multiple paths: inet-unicast
  ...
```

```
user@R8> show bgp neighbor 10.0.0.40
Peer: 10.0.0.40+179 AS 1       Local: 10.0.0.80+55416 AS 1
  Type: Internal    State: Established    Flags: <Sync>
  ...
  NLRI's for which peer can send multiple paths: inet-unicast
  ...
```

Verifying That Router R1 Is Advertising Multiple Paths

Purpose

Make sure that multiple paths to the 172.16.198.1/32 destination and multiple paths to the 172.16.199.1/32 destination are advertised to Router R4.

Action

```
user@R1> show route advertising-protocol bgp 10.0.0.40
inet.0: 21 destinations, 25 routes (21 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.200.0/30            10.0.0.20                    100        2 I
```

Meaning

When you see one prefix and more than one next hop, it means that multiple paths are advertised to Router R4.

Verifying That Router R4 Is Receiving and Advertising Multiple Paths

Purpose

Make sure that multiple paths to the 172.16.199.1/32 destination are received from Router R1 and advertised to Router R8. Make sure that multiple paths to the 172.16.198.1/32 destination are received from Router R1, but only one path to this destination is advertised to Router R8.

Action

```
user@R4> show route receive-protocol bgp 10.0.0.10
inet.0: 19 destinations, 22 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
```

```
                        10.0.0.30                100        2 I
                        10.0.15.2                100        2 2 I
* 172.16.200.0/30          10.0.0.20                100        2 I
```

```
user@R4> show route advertising-protocol bgp 10.0.0.80
inet.0: 19 destinations, 22 routes (19 active, 0 holddown, 0 hidden)
  Prefix                 Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32           10.0.15.2                    100        2 2 I
* 10.0.0.60/32           10.0.0.20                    100        2 I
* 10.0.0.70/32           10.0.0.30                    100        2 I
* 172.16.198.1/32           10.0.0.20                100        2 I
* 172.16.199.1/32           10.0.0.20                100        2 I
                         10.0.0.30                    100        2 I
                         10.0.15.2                    100        2 2 I
* 172.16.200.0/30           10.0.0.20                100        2 I
```

Meaning

The `show route receive-protocol` command shows that Router R4 receives two paths to the 172.16.198.1/32 destination and three paths to the 172.16.199.1/32 destination. The `show route advertising-protocol` command shows that Router R4 advertises only one path to the 172.16.198.1/32 destination and advertises all three paths to the 172.16.199.1/32 destination.

Because of the prefix policy that is applied to Router R4, Router R4 does not advertise multiple paths to the 172.16.198.1/32 destination. Router R4 advertises only one path to the 172.16.198.1/32 destination even though it receives multiple paths to this destination.

Verifying That Router R8 Is Receiving Multiple Paths

Purpose

Make sure that Router R8 receives multiple paths to the 172.16.199.1/32 destination through Router R4. Make sure that Router R8 receives only one path to the 172.16.198.1/32 destination through Router R4.

Action

```
user@R8> show route receive-protocol bgp 10.0.0.40
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 200.1.1.0/30            10.0.0.20                    100        2 I
```

Checking the Path ID

Purpose

On the downstream devices, Router R4 and Router R8, verify that a path ID uniquely identifies the path. Look for the `Addpath Path ID:` string.

Action

```
user@R4> show route 172.16.199.1/32 detail

inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
172.16.199.1/32 (3 entries, 3 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 9
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.0.20
                Indirect next hop: 92041c8 262146
                State: <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (3): 2-KRT 3-BGP RT Background 4-Resolve tree 1
```

```
                AS path: 2 I (Originator) Cluster list:  10.0.0.10
                AS path:  Originator ID: 10.0.0.20
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 1
        BGP     Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.0.30
                Indirect next hop: 92042ac 262151
                State: <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (1): 3-BGP RT Background
                AS path: 2 I (Originator) Cluster list:  10.0.0.10
                AS path:  Originator ID: 10.0.0.30
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 2
        BGP     Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.15.2
                Indirect next hop: 92040e4 262150
                State: <Int Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (1): 3-BGP RT Background
                AS path: 2 2 I
                Accepted
                Localpref: 100
```

```
                Router ID: 10.0.0.10
                Addpath Path ID: 3
```

```
user@R8> show route 172.16.199.1/32 detail

inet.0: 17 destinations, 19 routes (17 active, 0 holddown, 0 hidden)
172.16.199.1/32 (3 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 9
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.0.20
                Indirect next hop: 91fc0e4 262148
                State: <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                Announcement bits (2): 2-KRT 4-Resolve tree 1
                AS path: 2 I (Originator) Cluster list:  10.0.0.40 10.0.0.10
                AS path:  Originator ID: 10.0.0.20
                Accepted
                Localpref: 100
                Router ID: 10.0.0.40
                Addpath Path ID: 1
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.0.30
                Indirect next hop: 91fc1c8 262152
                State: <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                AS path: 2 I (Originator) Cluster list:  10.0.0.40 10.0.0.10
                AS path:  Originator ID: 10.0.0.30
```

```
                Accepted
                Localpref: 100
                Router ID: 10.0.0.40
                Addpath Path ID: 2
          BGP   Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.15.2
                Indirect next hop: 91fc2ac 262153
                State: <Int Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                AS path: 2 2 I (Originator) Cluster list:  10.0.0.40
                AS path:  Originator ID: 10.0.0.10
                Accepted
                Localpref: 100
                Router ID: 10.0.0.40
                Addpath Path ID: 3
```

SEE ALSO

Understanding the Advertisement of Multiple Paths to a Single Destination in BGP

Understanding Adding AS Numbers to BGP AS Paths

Example: Configuring Selective Advertising of BGP Multiple Paths for Load Balancing

IN THIS SECTION

- Requirements | 650
- Overview | 650

Configuration | 651
Verification | 661

This example shows how to configure selective advertising of BGP multiple paths. Advertising all available multiple paths might result in a large overhead of processing on device memory and is a scaling consideration, too. You can configure a BGP route reflector to advertise only contributor multipaths for load balancing.

Requirements

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

This example uses the following hardware and software components:

- Eight routers that can be a combination of M Series, MX Series, or T Series routers
- Junos OS Release 16.1R2 or later on the device

Overview

IN THIS SECTION

Topology | 650

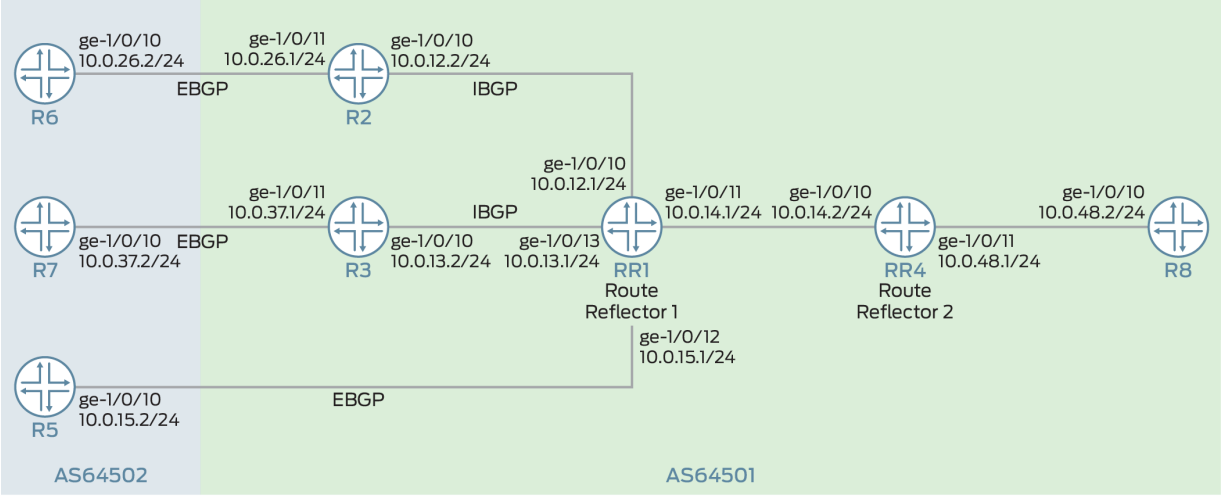
Beginning with Junos OS Release 16.1R2, you can restrict BGP `add-path` to advertise contributor multiple paths only. You can limit and configure up to six prefixes that the BGP `multipath` algorithm selects. Selective advertising of multiple paths facilitates Internet service providers and data centers that use route reflector to build in-path diversity in IBGP. You can enable a BGP route reflector to advertise multipaths that are contributor paths for load balancing.

Topology

In Figure 48 on page 651, RR1 and RR4 are route reflectors. Router R2 and R3 are clients to the route reflector RR1. Router R8 is a client to route reflector RR4. The RR1 group with neighbors R2 and R3 is configured for multipath. Routers R5, R6, and Router R7 redistribute static routes 199.1.1.1/32 and 198.1.1.1/32 into BGP.

A load-balancing policy is configured at Router RR1 such that the 199.1.1.1/32 routes have multipath calculated. The multipath feature is configured under add-path for neighbor RR4. However, Router RR4 does not have load-balancing multipath configured. Router RR1 is configured to send Router RR4 up to six add path routes to 199.1.1.1/32 chosen from multipath candidate routes.

Figure 48: Example: Configuring Selective Advertising of BGP Multiple Paths for Load Balancing



Configuration

IN THIS SECTION

- CLI Quick Configuration | 652
- Configuring Router RR1 | 656
- Results | 658

CLI Quick Configuration

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

Router RR1

```
set interfaces ge-1/0/10 unit 0 description RR1->R2
set interfaces ge-1/0/10 unit 0 family inet address 10.0.12.1/24
set interfaces ge-1/0/11 unit 0 description RR1->RR4
set interfaces ge-1/0/11 unit 0 family inet address 10.0.14.1/24
set interfaces ge-1/0/12 unit 0 description RR1->R5
set interfaces ge-1/0/12 unit 0 family inet address 10.0.15.1/24
set interfaces ge-1/0/13 unit 0 description RR1->R3
set interfaces ge-1/0/13 unit 0 family inet address 10.0.13.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.10/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.10
set protocols bgp group rr cluster 10.0.0.10
set protocols bgp group rr multipath
set protocols bgp group rr neighbor 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.30
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.2 local-address 10.0.15.1
set protocols bgp group e1 neighbor 10.0.15.2 peer-as 64502
set protocols bgp group rr_rr type internal
set protocols bgp group rr_rr local-address 10.0.0.10
set protocols bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send path-count 6
set protocols bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send multipath
set protocols ospf area 0.0.0.0 interface lo0.10 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface ge-1/0/13
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set protocols ospf area 0.0.0.0 interface ge-1/0/12
set policy-options prefix-list match_199 199.1.1.1/32
set policy-options policy-statement loadbal_199 term match_100 from prefix-list match_199
set policy-options policy-statement loadbal_199 from route-filter 199.1.1.1/32 exact
set policy-options policy-statement loadbal_199 then load-balance per-packet
set routing-options router-id 10.0.0.10
set routing-options autonomous-system 64501
set routing-options forwarding-table export loadbal_199
```

Router R2

```
set interfaces ge-1/0/10 unit 0 description R2->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.12.2/24
set interfaces ge-1/0/11 unit 0 description R2->R6
set interfaces ge-1/0/11 unit 0 family inet address 10.0.26.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.20/32
set protocols bgp group rr local-address 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.2 peer-as 64502
set protocols ospf area 0.0.0.0 interface lo0.20 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 64501
```

Router R3

```
set interfaces ge-1/0/10 unit 0 description R3->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.13.2/24
set interfaces ge-1/0/11 unit 0 description R3->R7
set interfaces ge-1/0/11 unit 0 family inet address 10.0.37.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.30/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.30
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.2 peer-as 64502
set protocols ospf area 0.0.0.0 interface lo0.30 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface ge-1/0/13
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 64501
```

Router RR4

```
set interfaces ge-1/0/10 unit 0 description RR4->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.14.2/24
set interfaces ge-1/0/11 unit 0 description RR4->R8
set interfaces ge-1/0/11 unit 0 family inet address 10.0.48.1/24
```

```
set interfaces lo0 unit 0 family inet address 10.0.0.40/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.40
set protocols bgp group rr family inet unicast add-path receive
set protocols bgp group rr neighbor 10.0.0.10
set protocols bgp group rr_client type internal
set protocols bgp group rr_client local-address 10.0.0.40
set protocols bgp group rr_client cluster 10.0.0.40
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send prefix-
policy addpath-communities-send-4713-100
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send path-
count 2
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send multipath
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface lo0.40 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set policy-options prefix-list match_199 199.1.1.1/32
set routing-options autonomous-system 64501
```

Router R5

```
set interfaces ge-1/0/10 unit 0 description R5->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.15.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.50/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.1 export s2b
set protocols bgp group e1 neighbor 10.0.15.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then as-path-expand 2
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R6

```
set interfaces ge-1/0/10 unit 0 description R6->R2
set interfaces ge-1/0/10 unit 0 family inet address 10.0.26.2/24
```

```
set interfaces lo0 unit 0 family inet address 10.0.0.60/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.1 export s2b
set protocols bgp group e1 neighbor 10.0.26.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R7

```
set interfaces ge-1/0/10 unit 0 description R7->R3
set interfaces ge-1/0/10 unit 0 family inet address 10.0.37.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.70/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.1 export s2b
set protocols bgp group e1 neighbor 10.0.37.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R8

```
set interfaces ge-1/0/10 unit 0 description R8->RR4
set interfaces ge-1/0/10 unit 0 family inet address 10.0.48.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.80/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.80
set protocols bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10.8
set routing-options autonomous-system 64501
set chassis fpc 1 pic 0 tunnel-services bandwidth 1g
```

Configuring Router RR1

Step-by-Step Procedure

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

To configure Router RR1:

NOTE: Repeat this procedure for other routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv4 addresses.

```
[edit interfaces]
user@RR1# set ge-1/0/10 unit 0 description RR1->R2
user@RR1# set ge-1/0/10 unit 0 family inet address 10.0.12.1/24
user@RR1# set ge-1/0/11 unit 0 description RR1->RR4
user@RR1# set ge-1/0/11 unit 0 family inet address 10.0.14.1/24
user@RR1# set ge-1/0/12 unit 0 description RR1->R5
user@RR1# set ge-1/0/12 unit 0 family inet address 10.0.15.1/24
user@RR1# set ge-1/0/13 unit 0 description RR1->R3
user@RR1# set ge-1/0/13 unit 0 family inet address 10.0.13.1/24
```

2. Configure the loopback address.

```
[edit interfaces]
user@RR1# set lo0 unit 0 family inet address 10.0.0.10/32
```

3. Configure interior gateway protocol (IGP) such as OSPF or IS-IS.

```
[edit protocols]
user@RR1# set ospf area 0.0.0.0 interface lo0.10 passive
user@RR1# set ospf area 0.0.0.0 interface ge-1/0/10
user@RR1# set ospf area 0.0.0.0 interface ge-1/0/13
user@RR1# set ospf area 0.0.0.0 interface ge-1/0/11
user@RR1# set ospf area 0.0.0.0 interface ge-1/0/12
```

4. Configure internal group rr for interfaces connecting to internal routers R2 and R3.

```
[edit protocols]
user@RR1# set bgp group rr type internal
user@RR1# set bgp group rr local-address 10.0.0.10
user@RR1# set bgp group rr cluster 10.0.0.10
user@RR1# set bgp group rr neighbor 10.0.0.20
user@RR1# set bgp group rr neighbor 10.0.0.30
```

5. Configure load balancing for internal BGP group rr.

```
[edit protocols]
user@RR1# set bgp group rr multipath
```

6. Configure internal group rr_rr for route reflectors.

```
[edit protocols]
user@RR1# set bgp group rr_rr type internal
user@RR1# set bgp group rr_rr local-address 10.0.0.10
```

7. Configure the addpath multipath feature to advertise contributor multiple paths only and limit the number of advertised multipaths to 6.

```
[edit protocols]
user@RR1# set bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send multipath
user@RR1# set bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send path-
count 6
```

8. Configure EBGP on interfaces connecting to the external edge routers.

```
[edit protocols]
user@RR1# set bgp group e1 type external
user@RR1# set bgp group e1 neighbor 10.0.15.2 local-address 10.0.15.1
user@RR1# set bgp group e1 neighbor 10.0.15.2 peer-as 64502
```

9. Define a policy loadbal_199 for per packet load balancing.

```
[edit policy-options]
user@RR1# set prefix-list match_199 199.1.1.1/32
user@RR1# set policy-statement loadbal_199 term match_100 from prefix-list match_199
user@RR1# set policy-statement loadbal_199 from route-filter 199.1.1.1/32 exact
user@RR1# set policy-statement loadbal_199 then load-balance per-packet
```

10. Apply the defined export policy loadbal_199.

```
[edit routing-options]
user@RR1# set forwarding-table export loadbal_199
```

11. Configure the router ID and the autonomous system for BGP hosts.

```
[edit routing-options]
user@RR1# set router-id 10.0.0.10
user@RR1# set autonomous-system 64501
```

Results

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

```
[edit]
user@RR1# show interfaces
ge-1/0/10 {
    unit 0 {
        description RR1->R2;
        family inet {
            address 10.0.12.1/24;
        }
    }
}
ge-1/0/11 {
    unit 0 {
        description RR1->RR4;
        family inet {
```

```
            address 10.0.14.1/24;
        }
    }
}
ge-1/0/12 {
    unit 0 {
        description RR1->R5;
        family inet {
            address 10.0.15.1/24;
        }
    }
}
ge-1/0/13 {
    unit 0 {
        description RR1->R3;
        family inet {
            address 10.0.13.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.0.0.10/32;
        }
    }
}
```

```
[edit]
user@RR1# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.10;
        cluster 10.0.0.10;
        multipath;
        neighbor 10.0.0.20;
        neighbor 10.0.0.30;
    }
    group e1 {
        type external;
```

```
        neighbor 10.0.15.2 {
            local-address 10.0.15.1;
            peer-as 64502;
        }
    }
    group rr_rr {
        type internal;
        local-address 10.0.0.10;
        neighbor 10.0.0.40 {
            family inet {
                unicast {
                    add-path {
                        send {
                            path-count 6;
                            multipath;
                        }
                    }
                }
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface lo0.10 {
            passive;
        }
        interface ge-1/0/10;
        interface ge-1/0/13;
        interface ge-1/0/11;
        interface ge-1/0/12;
    }
}
```

```
[edit]
user@RR1# show routing-options
router-id 10.0.0.10;
```

```
autonomous-system 64501;
forwarding-table {
    export load-bal_199;
}
```

```
[edit]
user@RR1# show policy-options
prefix-list match_199 {
    199.1.1.1/32;
}
policy-statement loadbal_199 {
    term match_100 {
        from {
            prefix-list match_199;
        }
    }
    from {
        route-filter 199.1.1.1/32 exact;
    }
    then {
        load-balance per-packet;
    }
}
```

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

```
user@RR1# commit
```

Verification

IN THIS SECTION

- Verifying the Multipath Routes for the Static Route 199.1.1.1/32 | **662**
- Verifying That the Multipath Routes are Advertised from Router RR1 to Router RR4 | **664**
- Verifying that Router RR4 Advertises One Route for 199.1.1.1/32 to Router R8 | **665**

Confirm that the configuration is working properly.

Verifying the Multipath Routes for the Static Route 199.1.1.1/32

Purpose

Verify the available multipath routes for destination 199.1.1.1/32.

Action

From operational mode, run the **show route 199.1.1.1/32 detail** command on Router RR1.

```
user@RR1> show route 199.1.1.1/32 detail
inet.0: 22 destinations, 26 routes (22 active, 0 holddown, 0 hidden)
199.1.1.1/32 (3 entries, 2 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xae5cc90
                Next-hop reference count: 1
                Source: 10.0.0.20
                Next hop type: Router, Next hop index: 1118
                Next hop: 10.0.12.2 via lt-1/0/10.1, selected
                Session Id: 0x0
                Next hop type: Router, Next hop index: 1115
                Next hop: 10.0.13.2 via lt-1/0/10.9
                Session Id: 0x0
                Protocol next hop: 10.0.0.20
                Indirect next hop: 0xc409410 1048574 INH Session ID: 0x0
                Protocol next hop: 10.0.0.30
                Indirect next hop: 0xc409520 1048575 INH Session ID: 0x0
                State:  <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 4:03:29     Metric2: 1
                Validation State: unverified
                Task: BGP_1.10.0.0.20
                Announcement bits (3): 2-KRT 3-BGP_RT_Background 4-Resolve tree 2
                AS path: 2 I
                Communities: 4713:100
                Accepted Multipath
                Localpref: 100
                Router ID: 10.0.0.20
         BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xae0ec10
```

```
                Next-hop reference count: 4
                Source: 10.0.0.30
                Next hop type: Router, Next hop index: 1115
                Next hop: 10.0.13.2 via lt-1/0/10.9, selected
                Session Id: 0x0
                Protocol next hop: 10.0.0.30
                Indirect next hop: 0xc409520 1048575 INH Session ID: 0x0
                State: <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:  64501 Peer AS: 64501
                Age: 4:03:29     Metric2: 1
                Validation State: unverified
                Task: BGP_1.10.0.0.30
                Announcement bits (1): 3-BGP_RT_Background
                AS path: 2 I
                Communities: 4713:100
                Accepted MultipathContrib
                Localpref: 100
                Router ID: 10.0.0.30
        BGP     Preference: 170/-101
                Next hop type: Router, Next hop index: 1105
                Address: 0xae0e970
                Next-hop reference count: 5
                Source: 10.0.15.2
                Next hop: 10.0.15.2 via lt-1/0/10.6, selected
                Session Id: 0x0
                State: <Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     2
                Age: 4:05:01
                Validation State: unverified
                Task: BGP_2.10.0.15.2
                AS path: 2 2 I
                Communities: 4713:100
                Accepted
                Localpref: 100
                Router ID: 10.0.0.50
```

Meaning

The selective advertising multipath feature is enabled on Router RR1 and there is more than one nexthop available for route 199.1.1.1/32. The two available next hops for route 199.1.1.1/32 are 10.0.0.20 and 10.0.0.30.

Verifying That the Multipath Routes are Advertised from Router RR1 to Router RR4

Purpose

Verify that Router RR1 is advertising the multipath routes.

Action

From operational mode, run the **show route advertising-protocol bgp 10.0.0.40** command on Router RR1.

```
user@RR1> show route advertising-protocol bgp 10.0.0.40
inet.0: 22 destinations, 26 routes (22 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                   100        2 2 I
* 10.0.0.60/32            10.0.0.20                   100        2 I
* 10.0.0.70/32            10.0.0.30                   100        2 I
* 198.1.1.1/32            10.0.0.20                   100        2 I
* 199.1.1.1/32            10.0.0.20                   100        2 I
                          10.0.0.30                   100        2 I
```

```
user@RR1> show route advertising-protocol bgp 10.0.0.40 detail
inet.0: 22 destinations, 26 routes (22 active, 0 holddown, 0 hidden)
* 10.0.0.50/32 (1 entry, 1 announced)
 BGP group rr_rr type Internal
     Nexthop: 10.0.15.2
     Localpref: 100
     AS path: [1] 2 2 I
     Communities: 4713:100
     Addpath Path ID: 1
….* 199.1.1.1/32 (3 entries, 2 announced)
 BGP group rr_rr type Internal
```

```
     Nexthop: 10.0.0.20
     Localpref: 100
     AS path: [1] 2 I
     Communities: 4713:100
     Cluster ID: 10.0.0.10
     Originator ID: 10.0.0.20
     Addpath Path ID: 1
 BGP group rr_rr type Internal
     Nexthop: 10.0.0.30
     Localpref: 100
     AS path: [1] 2 I
     Communities: 4713:100
     Cluster ID: 10.0.0.10
     Originator ID: 10.0.0.30
     Addpath Path ID: 2
```

Meaning

Router RR1 is advertising two next hops 10.0.0.20 and 10.0.0.30 for route 199.1.1.1/32 to Router RR4.

Verifying that Router RR4 Advertises One Route for 199.1.1.1/32 to Router R8

Purpose

Multipath is not configured on Router RR4, therefore route 199.1.1.1/32 is not eligible for add-path. Verify that Router RR4 advertises only one route for 199.1.1.1/32 to Router R8.

Action

From operational mode, run the **show route advertising-protocol bgp 10.0.0.80** command on Router RR4.

```
user@RR4> show route advertising-protocol bgp 10.0.0.80 detail
inet.0: 20 destinations, 21 routes (20 active, 0 holddown, 0 hidden)
* 10.0.0.50/32 (1 entry, 1 announced)
 BGP group rr_client type Internal
     Nexthop: 10.0.15.2
     Localpref: 100
     AS path: [1] 2 2 I
     Communities: 4713:100
     Cluster ID: 10.0.0.40
```

```
     Originator ID: 10.0.0.10

     Addpath Path ID: 1
…
* 198.1.1.1/32 (1 entry, 1 announced)
 BGP group rr_client type Internal
     Nexthop: 10.0.0.20
     Localpref: 100
     AS path: [1] 2 I (Originator)
     Cluster list:  10.0.0.10
     Originator ID: 10.0.0.20
     Communities: 4713:100
     Cluster ID: 10.0.0.40
     Addpath Path ID: 1

* 199.1.1.1/32 (2 entries, 1 announced)
 BGP group rr_client type Internal
     Nexthop: 10.0.0.20
     Localpref: 100
     AS path: [1] 2 I (Originator)
     Cluster list:  10.0.0.10
     Originator ID: 10.0.0.20
     Communities: 4713:100
     Cluster ID: 10.0.0.40
     Addpath Path ID: 1
```

Meaning

Since multipath is not enabled on Router RR4, only one path 10.0.0.20 is advertised to Router R8.

SEE ALSO

send

Example: Configuring a Routing Policy to Select and Advertise Multipaths Based on BGP Community Value

IN THIS SECTION

- Requirements | 667
- Overview | 667
- Configuration | 669
- Verification | 677

Advertising all available multiple paths might result in a large overhead of processing on device memory. If you want to advertise a limited subset of prefixes without actually knowing the prefixes in advance, you can use the BGP community value to identify prefix routes that need to be advertised to BGP neighbors. This example shows how to define a routing policy to filter and advertise multiple paths based on a known BGP community value.

Requirements

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

This example uses the following hardware and software components:

- Eight routers that can be a combination of M Series, MX Series, or T Series routers
- Junos OS Release 16.1R2 or later on the device

Overview

IN THIS SECTION

- Topology | 668

Beginning with Junos OS 16.1R2, you can define a policy to identify eligible multiple path prefixes based on community values. BGP advertises these community-tagged routes in addition to the active path to a given destination. If the community value of a route does not match the community value defined in the policy, then BGP does not advertise that route. This feature allows BGP to advertise not more than 20

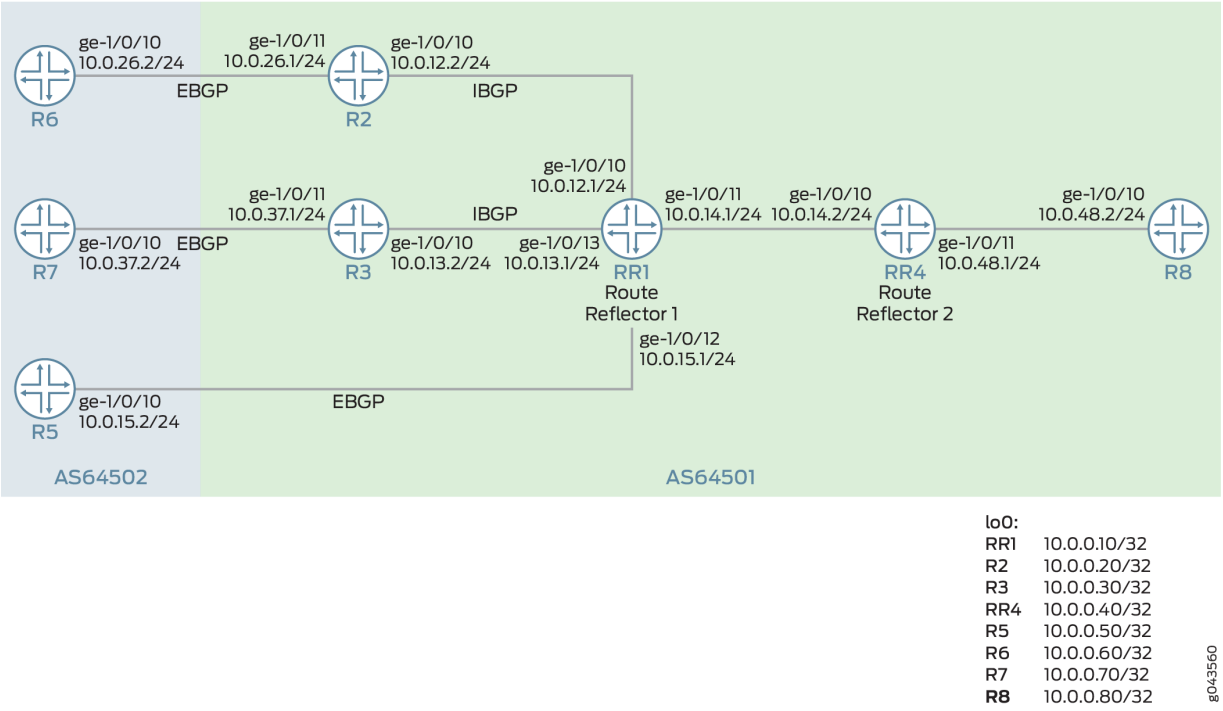
paths to a given destination. You can limit and configure the number of prefixes that BGP considers for multiple paths without actually knowing the prefixes in advance. Instead, a known BGP community value determines whether or not a prefix is advertised.

Topology

In Figure 49 on page 668, RR1 and RR4 are route reflectors. Router R2 and R3 are clients to the route reflector RR1. Router R8 is a client to route reflector RR4. Routers R5, R6, and Router R7 redistribute static routes into BGP. Router R5 advertises static routes 199.1.1.1/32 and 198.1.1.1/32 with community value 4713:100.

Router RR1 is configured to send up to six paths (per destination) to Router RR4. Router RR4 is configured to send up to six paths to Router R8. Router R8 is configured to receive multiple paths from Router RR4. The add-path community configuration restricts Router RR4 to send multiple paths for routes that contain only the 4713:100 community value. Router RR4 filters and advertises multipaths that contain only 4714:100 community value.

Figure 49: Example: Configuring BGP to Advertise Multipaths Based on Community Value



Configuration

IN THIS SECTION

- CLI Quick Configuration | 669
- Configuring Router RR4 | 673
- Results | 675

CLI Quick Configuration

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

Router RR1

```
set interfaces ge-1/0/10 unit 0 description RR1->R2
set interfaces ge-1/0/10 unit 0 family inet address 10.0.12.1/24
set interfaces ge-1/0/11 unit 0 description RR1->RR4
set interfaces ge-1/0/11 unit 0 family inet address 10.0.14.1/24
set interfaces ge-1/0/12 unit 0 description RR1->R5
set interfaces ge-1/0/12 unit 0 family inet address 10.0.15.1/24
set interfaces ge-1/0/13 unit 0 description RR1->R3
set interfaces ge-1/0/13 unit 0 family inet address 10.0.13.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.10/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.10
set protocols bgp group rr cluster 10.0.0.10
set protocols bgp group rr multipath
set protocols bgp group rr neighbor 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.30
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.2 local-address 10.0.15.1
set protocols bgp group e1 neighbor 10.0.15.2 peer-as 64502
set protocols bgp group rr_rr type internal
set protocols bgp group rr_rr local-address 10.0.0.10
set protocols bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send path-count 6
set protocols ospf area 0.0.0.0 interface lo0.10 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
```

```
set protocols ospf area 0.0.0.0 interface ge-1/0/13
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set protocols ospf area 0.0.0.0 interface ge-1/0/12
set routing-options router-id 10.0.0.10
set routing-options autonomous-system 64501
```

Router R2

```
set interfaces ge-1/0/10 unit 0 description R2->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.12.2/24
set interfaces ge-1/0/11 unit 0 description R2->R6
set interfaces ge-1/0/11 unit 0 family inet address 10.0.26.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.20/32
set protocols bgp group rr local-address 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.2 peer-as 64502
set protocols ospf area 0.0.0.0 interface lo0.20 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 64501
```

Router R3

```
set interfaces ge-1/0/10 unit 0 description R3->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.13.2/24
set interfaces ge-1/0/11 unit 0 description R3->R7
set interfaces ge-1/0/11 unit 0 family inet address 10.0.37.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.30/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.30
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.2 peer-as 64502
set protocols ospf area 0.0.0.0 interface lo0.30 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface ge-1/0/13
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 64501
```

Router RR4

```
set interfaces ge-1/0/10 unit 0 description RR4->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.14.2/24
set interfaces ge-1/0/11 unit 0 description RR4->R8
set interfaces ge-1/0/11 unit 0 family inet address 10.0.48.1/24
set interfaces lo0 unit 0 family inet address 10.0.0.40/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.40
set protocols bgp group rr family inet unicast add-path receive
set protocols bgp group rr neighbor 10.0.0.10
set protocols bgp group rr_client type internal
set protocols bgp group rr_client local-address 10.0.0.40
set protocols bgp group rr_client cluster 10.0.0.40
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send prefix-
policy addpath-communities-send-4713-100
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send path-
count 6
set protocols ospf area 0.0.0.0 interface ge-1/0/10
set protocols ospf area 0.0.0.0 interface lo0.40 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/11
set policy-options community addpath-communities-send members 4713:100
set policy-options policy-statement addpath-communities-send-4713-100 term term1 from protocol
bgp
set policy-options policy-statement addpath-communities-send-4713-100 term term1 from community
addpath-communities-send
set policy-options policy-statement addpath-communities-send-4713-100 term term1 then add-path
send-count 16
set policy-options policy-statement addpath-communities-send-4713-100 term term1 then accept
set routing-options autonomous-system 64501
```

Router R5

```
set interfaces ge-1/0/10 unit 0 description R5->RR1
set interfaces ge-1/0/10 unit 0 family inet address 10.0.15.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.50/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.1 export s2b
set protocols bgp group e1 neighbor 10.0.15.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
```

```
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then as-path-expand 2
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R6

```
set interfaces ge-1/0/10 unit 0 description R6->R2
set interfaces ge-1/0/10 unit 0 family inet address 10.0.26.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.60/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.1 export s2b
set protocols bgp group e1 neighbor 10.0.26.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R7

```
set interfaces ge-1/0/10 unit 0 description R7->R3
set interfaces ge-1/0/10 unit 0 family inet address 10.0.37.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.70/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.1 export s2b
set protocols bgp group e1 neighbor 10.0.37.1 peer-as 64501
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then community add addpath-community
set policy-options policy-statement s2b then accept
set policy-options community addpath-community members 4713:100
set routing-options static route 199.1.1.1/32 reject
set routing-options autonomous-system 64502
```

Router R8

```
set interfaces ge-1/0/10 unit 0 description R8->RR4
set interfaces ge-1/0/10 unit 0 family inet address 10.0.48.2/24
set interfaces lo0 unit 0 family inet address 10.0.0.80/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.80
set protocols bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/10.8
set routing-options autonomous-system 64501
set chassis fpc 1 pic 0 tunnel-services bandwidth 1g
```

Configuring Router RR4

Step-by-Step Procedure

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

To configure Router RR4:

NOTE: Repeat this procedure for other routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv4 addresses.

```
[edit interfaces]
user@RR4# set ge-1/0/10 unit 0 description RR4->RR1
user@RR4# set ge-1/0/10 unit 0 family inet address 10.0.14.2/24
user@RR4# set ge-1/0/11 unit 0 description RR4->R8
user@RR4# set ge-1/0/11 unit 0 family inet address 10.0.48.1/24
```

2. Configure the loopback address.

```
[edit interfaces]
user@RR4# set lo0 unit 0 family inet address 10.0.0.40/32
```

3. Configure OSPF or any other interior gateway protocol (IGP).

```
[edit protocols]
user@RR4# set ospf area 0.0.0.0 interface lo0.40 passive
user@RR4# set ospf area 0.0.0.0 interface ge-1/0/10
user@RR4# set ospf area 0.0.0.0 interface ge-1/0/11
```

4. Configure two IBGP groups rr for route reflectors and rr_client for clients of route reflectors.

```
[edit protocols]
user@RR4# set bgp group rr type internal
user@RR4# set bgp group rr local-address 10.0.0.40
user@RR4# set bgp group rr family inet unicast add-path receive
user@RR4# set bgp group rr neighbor 10.0.0.10
user@RR4# set bgp group rr_client type internal
user@RR4# set bgp group rr_client local-address 10.0.0.40
user@RR4# set bgp group rr_client cluster 10.0.0.40
```

5. Configure the feature to send multiple paths that contain 4713:100 community value only and limit the number of advertised multipaths to 6.

```
[edit protocols]
user@RR4# set bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send prefix-
policy addpath-communities-send-4713-100
user@RR4# set bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send path-
count 6
```

6. Define a policy `addpath-community-members 4713:100` to filter prefixes with the community value 4713:100 and restrict the device to send up to 16 paths to Router R8. This limit overrides the previously configured add-path send path-count of 6 at the BGP group hierarchy level.

```
[edit policy-options]
user@RR4# set community addpath-communities-send members 4713:100
user@RR4# set policy-statement addpath-communities-send-4713-100 term term1 from protocol bgp
user@RR4# set policy-statement addpath-communities-send-4713-100 term term1 from community
addpath-communities-send
user@RR4# set policy-statement addpath-communities-send-4713-100 term term1 then add-path
```

```
send-count 16
user@RR4# set policy-statement addpath-communities-send-4713-100 term term1 then accept
```

7. Configure the router ID and the autonomous system for BGP hosts.

```
[edit routing-options]
user@RR4# set router-id 10.0.0.40
user@RR4# set autonomous-system 64501
```

Results

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

```
[edit]
user@RR4# show interfaces
ge-1/0/10 {
    unit 0 {
        description RR4->RR1;
        family inet {
            address 10.0.14.2/24;
        }
    }
}
ge-1/0/11 {
    unit 0 {
        description RR4->R8;
        family inet {
            address 10.0.48.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.0.0.10/32;
        }
```

```
    }
}
```

```
[edit]
user@RR4# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.40;
        family inet {
            unicast {
                add-path {
                    receive;
                }
            }
        }
        neighbor 10.0.0.10;
    }
    group rr_client {
        type internal;
        local-address 10.0.0.40;
        cluster 10.0.0.40;
        neighbor 10.0.0.80 {
            family inet {
                unicast {
                    add-path {
                        send {
                            prefix-policy addpath-communities-send-4713-100;
                            path-count 6;
                        }
                    }
                }
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/10.0;
        interface lo0.40 {
            passive;
```

```
        }
        interface ge-1/0/11.0;
    }
}
```

```
[edit]
user@RR4# show policy-options
policy-statement addpath-communities-send-4713-100 {
    term term1 {
        from {
            protocol bgp;
            community addpath-communities-send;
        }
        then {
            add-path send-count 16;
            accept;
        }
    }
}
community addpath-communities-send members 4713:100;
```

```
[edit]
user@RR4# show routing-options
router-id 10.0.0.40;
autonomous-system 64501;
```

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

```
user@RR4# commit
```

Verification

IN THIS SECTION

- Verifying That the Multipath Routes are Advertised from Router RR4 to Router R8 | **678**
- Verifying That Router R8 Receives the Multipath Routes That Router RR4 Advertises | **678**

- Verifying That Router RR4 is Advertising only Multipath Routes with Community Value 4713:100 to Router R8 | **679**

Confirm that the configuration is working properly.

Verifying That the Multipath Routes are Advertised from Router RR4 to Router R8

Purpose

Verify that Router RR4 can send multiple paths to Router R8.

Action

From operational mode, run the **show route advertising-protocol bgp** ***neighbor-address*** command on Router RR4.

```
user@RR4> show route advertising-protocol bgp 10.0.0.80
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 198.1.1.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 199.1.1.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
```

Meaning

Router RR4 is advertising multiple paths 10.0.0.20, 10.0.0.30, and 10.0.15.2 to Router R8.

Verifying That Router R8 Receives the Multipath Routes That Router RR4 Advertises

Purpose

Verify that Router R8 is receiving the multipath routes from Router RR4.

Action

From operational mode, run the **show route receive-protocol bgp *neighbor-address*** command on Router R8.

```
user@R8> show route receive-protocol bgp 10.0.0.40

inet.0: 19 destinations, 22 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 198.1.1.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 199.1.1.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
```

Meaning

Router R8 is receiving multiple next hops 10.0.0.20, 10.0.0.30, and 10.0.15.2 for route 199.1.1.1/32 from Router RR4.

Verifying That Router RR4 is Advertising only Multipath Routes with Community Value 4713:100 to Router R8

Purpose

Router RR4 must advertise multipath routes with community value of 4713:100 only to Router R8.

Action

From operational mode, run the **show route 199.1.1.1/32 detail** command on Router RR4.

```
user@RR4> show route 199.1.1.1/32 detail
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
199.1.1.1/32 (3 entries, 3 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
```

```
                Address: 0xae0ea90
                Next-hop reference count: 6
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 1115
                Next hop: 10.0.14.1 via ge-1/0/10.4, selected
                Session Id: 0x0
                Protocol next hop: 10.0.0.20
                Indirect next hop: 0xc4091f0 1048581 INH Session ID: 0x0
                State: <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 4d 20:56:53        Metric2: 2
                Validation State: unverified
                Task: BGP_1.10.0.0.10
                Announcement bits (3): 2-KRT 3-BGP_RT_Background 4-Resolve tree 2
                AS path: 2 I (Originator)
                Cluster list:  10.0.0.10
                Originator ID: 10.0.0.20
                Communities: 4713:100
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 1
        BGP     Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xae0eb50
                Next-hop reference count: 3
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 1115
                Next hop: 10.0.14.1 via lt-1/0/10.4, selected
                Session Id: 0x0
                Protocol next hop: 10.0.0.30
                Indirect next hop: 0xc409300 1048582 INH Session ID: 0x0
                State:  <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:     1 Peer AS:     1
                Age: 4d 20:56:53        Metric2: 2
                Validation State: unverified
                Task: BGP_1.10.0.0.10
                Announcement bits (1): 3-BGP_RT_Background
                AS path: 2 I (Originator)
                Cluster list:  10.0.0.10
                Originator ID: 10.0.0.30
                Communities: 4713:100
```

```
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 2
          BGP   Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xae0e9d0
                Next-hop reference count: 4
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 1115
                Next hop: 10.0.14.1 via lt-1/0/10.4, selected
                Session Id: 0x0
                Protocol next hop: 10.0.15.2
                Indirect next hop: 0xc4090e0 1048580 INH Session ID: 0x0
                State:  <Int Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     1
                Age: 4d 20:56:53        Metric2: 2
                Validation State: unverified
                Task: BGP_1.10.0.0.10
                Announcement bits (1): 3-BGP_RT_Background
                AS path: 2 2 I
                Communities: 4713:100
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 3
```

Meaning

Router RR4, is advertising three paths with community value of 4713:100 to Router R8.

SEE ALSO

send

Example: Configuring Selective Advertising of BGP Multiple Paths for Load Balancing | 649

Understanding BGP Multipath | 551

Configuring Recursive Resolution over BGP Multipath

Starting in Junos OS Release 17.3R1, when a BGP prefix that has a single protocol next hop is resolved over another BGP prefix that has multiple resolved paths (unilist), all the paths are selected for protocol next-hop resolution. In earlier Junos OS releases, only one of the paths is picked for protocol next-hop resolution because the resolver did not support load-balancing across all paths of the IBGP multipath route. The resolver in the routing protocol process (rpd) resolves the protocol next-hop address (PNH) into immediate forwarding next hops. The BGP recursive resolution feature enhances the resolver to resolve routes over IBGP multipath route and use all the feasible paths as next hops. This feature benefits densely connected networks where BGP is used to establish infrastructure connectivity such as WAN networks with high equal-cost multipath and seamless MPLS topology.

Before you begin configuring recursive resolution of BGP multipath, you must do the following:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and LDP.
4. Configure BGP.

To configure recursive resolution over multipath,

1. Define a policy that includes the `multipath-resolve` action .

```
[edit policy-options policy-statement policy-name then]
user@host# set multipath-resolve
```

2. Import the policy to resolve all the available paths of IBGP multipath route.

```
[edit routing-options resolution rib rib-name]
user@host# set import policy-name
```

3. Verify that BGP is resolving multipaths recursively and multiple next hops are available for load balancing traffic.

   From operational mode, enter the `show route resolution detail` command:

```
user@host> show route resolution detail 10.1.1.2
Tree Index: 1, Nodes 36, Reference Count 3
Contributing routing tables: inet.0 inet.3
Policy: [ abc ]
```

```
10.1.1.2/32 Originating RIB: inet.0
  Node path count: 1
  Next hop subtype: INDIRECT
  Indirect next hops: 2
        Protocol next hop: 10.1.1.1
        Inode flags: 0x206 path flags: 0x08
        Path fnh link: 0xc9321c0 path inh link: 0x0
        Indirect next hop: 0xb2b20f0 1048574 INH Session ID: 0x143
        Indirect path forwarding next hops: 1
                Next hop type: Router
                Next hop: 12.1.1.2 via ge-2/0/1.0
                Session Id: 0x144
                Next hop: 13.1.1.2 via ge-2/0/2.0
                Session Id: 0x145

    10.1.1.1/32 Originating RIB: inet.0
      Node path count: 1
  Node flags: 1
      Forwarding nexthops: 1 (Merged)
        Nexthop: 12.1.1.2 via ge-2/0/1.0

        Nexthop: 13.1.1.2 via ge-2/0/2.0


user@host> show route 10.1.1.2 extensive
inet.0: 37 destinations, 37 routes (36 active, 0 holddown, 1 hidden)
10.1.1.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.1.1.2/32 -> {indirect(1048574)}
        *Static Preference: 5
                Next hop type: Indirect, Next hop index: 0
                Address: 0xb39d1b0
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 581
                Next hop: 12.1.1.2 via ge-2/0/1.0, selected
                Session Id: 0x144
                Next hop: 13.1.1.2 via ge-2/0/2.0, selected
                Session Id: 0x145
                Protocol next hop: 10.1.1.1
                Indirect next hop: 0xb2b20f0 1048574 INH Session ID: 0x143
                State: <Active Int Ext>
                Age: 2:53     Metric2: 0
                Validation State: unverified
```

```
                Task: RT
                Announcement bits (2): 0-KRT 2-Resolve tree 1
                AS path: I
                Indirect next hops: 1
                        Protocol next hop: 10.1.1.1
                        Indirect next hop: 0xb2b20f0 1048574 INH Session ID: 0x143
                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 12.1.1.2 via ge-2/0/1.0
                                Session Id: 0x144
                                Next hop: 13.1.1.2 via ge-2/0/2.0
                                Session Id: 0x145
                                       10.1.1.1/32 Originating RIB: inet.0
                                 Node path count: 1
                          Node flags: 1
                                 Forwarding nexthops: 2 (Merged)
                                 Nexthop: 12.1.1.2 via ge-2/0/1.0
                                     Nexthop: 13.1.1.2 via ge-2/0/2.0
```

SEE ALSO

> *policy-statement*
>
> *show route resolution*

Configuring ECMP Next Hops for RSVP and LDP LSPs for Load Balancing

The Junos OS supports configurations of 16, 32, 64, or 128 equal-cost multipath (ECMP) next hops for RSVP and LDP LSP.s. For networks with high-volume traffic, this provides more flexibility to load-balance the traffic over as many as 128 LSPs.

To configure the maximum limit for ECMP next hops, include the `maximum-ecmp` *`next-hops`* statement at the `[edit chassis]` hierarchy level:

```
[edit chassis]
maximum-ecmp next-hops;
```

You can configure a maximum ECMP next-hop limit of 16, 32, 64, or 128 using this statement. The default limit is 16.

NOTE: MX Series routers with one or more Modular Port Concentrator (MPC) cards and with Junos OS 11.4 or earlier installed, support the configuration of the `maximum-ecmp` statement with only 16 next hops. You should *not* configure the `maximum-ecmp` statement with 32 or 64 next hops. When you commit the configuration with 32 or 64 next hops, the following warning message appears:

```
Error: Number of members in Unilist NH exceeds the maximum supported 16 on Trio.
```

The following types of routes support the ECMP maximum next-hop configuration for as many as 128 ECMP gateways:

- Static IPv4 and IPv6 routes with direct and indirect next-hop ECMPs
- LDP ingress and transit routes learned through associated IGP routes
- RSVP ECMP next hops created for LSPs
- OSPF IPv4 and IPv6 route ECMPs
- IS-IS IPv4 and IPv6 route ECMPs
- EBGP IPv4 and IPv6 route ECMPs
- IBGP (resolving over IGP routes) IPv4 and IPv6 route ECMPs

The enhanced ECMP limit of up to 128 ECMP next hops is also applicable for Layer 3 VPNs, Layer 2 VPNs, Layer 2 circuits, and VPLS services that resolve over an MPLS route, because the available ECMP paths in the MPLS route can also be used by such traffic.

NOTE: If RSVP LSPs are configured with bandwidth allocation, for ECMP next hops with more than 16 LSPs, traffic is not distributed optimally based on bandwidths configured. Some LSPs with smaller allocated bandwidths receive more traffic than the ones configured with higher bandwidths. Traffic distribution does not strictly comply with the configured bandwidth allocation. This caveat is applicable to the following routers:

- MX Series routers with all types of FPCs and DPCs, excluding MPCs. This caveat is *not* applicable to MX Series routers with line cards based on the Junos Trio chipset.

To view the details of the ECMP next hops, issue the `show route` command. The `show route summary command` also shows the current configuration for the maximum ECMP limit. To view details of the ECMP LDP paths, issue the `traceroute mpls ldp` command.

SEE ALSO

maximum-ecmp

Configuring Consistent Load Balancing for ECMP Groups

Per-packet load balancing allows you to spread traffic across multiple equal-cost paths. By default, when a failure occurs in one or more paths, the hashing algorithm recalculates the next hop for all paths, typically resulting in the redistribution of all flows. *Consistent load balancing* enables you to override this behavior so that only flows for links that are inactive are redirected. All existing active flows are maintained without disruption. In a data center environment, the redistribution of all flows when a link fails potentially results in significant traffic loss or a loss of service to servers whose links remain active. Consistent load balancing maintains all active links and instead remaps only those flows affected by one or more link failures. This feature ensures that flows connected to links that remain active continue uninterrupted.

This feature applies to topologies where members of an equal-cost multipath (ECMP) group are external BGP neighbors in a single-hop BGP session. Consistent load balancing does not apply when you add a new ECMP path or modify an existing path in any way. To add a new path with minimal disruption, define a new ECMP group without modifying the existing paths. In this way, clients can be moved to the new group gradually without terminating existing connections.

- (On MX Series) Only Modular Port Concentrators (MPCs) are supported.
- Both IPv4 and IPv6 paths are supported.
- ECMP groups that are part of a virtual routing and forwarding (VRF) instance or other routing instance are also supported.
- Multicast traffic is not supported.
- Aggregated interfaces are supported, but consistent load balancing is not supported among members of the link aggregation (LAG) bundle. Traffic from active members of the LAG bundle might be moved to another active member when one or more member links fail. Flows are rehashed when one or more LAG member links fail.
- We strongly recommend that you apply consistent load balancing to no more than a maximum of 1,000 IP prefixes per router or switch.
- Layer 3 adjacency over integrated routing and bridging (IRB) interfaces is supported.

You can configure the BGP add-path feature to enable replacement of a failed path with a new active path when one or more paths in the ECMP group fail. Configuring replacement of failed paths ensures that traffic flow on the failed paths only are redirected. Traffic flow on active paths will remain unaltered.

NOTE:

- When you configure consistent load balancing on generic routing encapsulation (GRE) tunnel interfaces, you must specify the inet address of the far end GRE interface so that the Layer 3 adjacencies over the GRE tunnel interfaces are installed correctly in the forwarding table. However, ECMP fast reroute (FRR) over GRE tunnel interfaces is not supported during consistent load balancing. You can specify the destination address on the router configured with consistent load balancing at the `[edit interfaces interface name unit unit name family inet address address]` hierarchy level. For example:

  ```
  [edit interfaces]
  user@host# set interfaces gr-4/0/0 unit 21 family inet address 10.10.31.2/32
  destination 10.10.31.1
  ```

  For more information on generic routing encapsulation see *Configuring Generic Routing Encapsulation Tunneling*.

- Consistent load balancing does not support BGP multihop for EBGP neighbors. Therefore, do not enable the `multihop` option on devices configured with consistent load balancing.

To configure consistent load balancing for ECMP groups:

1. Configure BGP and enable the BGP group of external peers to use multiple paths.
2. Create a routing policy to match incoming routes to one or more destination prefixes.

   ```
   [edit policy-options]
   user@host# set policy-statement policy-statement-name from route-filter destination-prefix
   orlonger
   ```

3. Apply consistent load balancing to the routing policy so that only traffic flows to one or more destination prefixes that experience a link failure are redirected to an active link.

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

4. Create a separate routing policy and enable per-packet load balancing.

NOTE: You must configure and apply a per-packet load-balancing policy to install all routes in the forwarding table.

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

5. Apply the routing policy for consistent load balancing to the BGP group of external peers.

NOTE: Consistent load balancing can be applied only to BGP external peers. This policy cannot be applied globally.

```
[edit protocols bgp]
user@host# set group group-name  import policy-statement-name
#This policy-statement-name refers to the policy created in Step 2.
```

6. (Optional) Enable bidirectional forwarding detection (BFD) for each external BGP neighbor.

```
[edit protocols bgp]
user@host# set group group-name neighbor ip-address bfd-liveness-detection milliseconds
```

NOTE: This step shows the minimum BFD configuration required. You can configure additional options for BFD.

7. Apply the per-prefix load-balancing policy globally to install all next-hop routes in the forwarding table.

```
[edit routing-options]
user@host# set forwarding table export policy-statement-name
#This policy-statement-name refers to the policy created in Step 4.
```

8. (Optional) Enable fast reroute for ECMP routes.

```
[edit routing-options]
user@host# set forwarding-table ecmp-fast-reroute
```

9. Verify the status of one or more ECMP routes for which you enabled consistent load balancing.

```
user@host> show route destination-prefix extensive
```

The output of the command displays the following flag when consistent load balancing is enabled:

```
State: <Active Ext LoadBalConsistentHash>
```

Improve Network Resiliency Using Multiple ECMP BGP Peers

IN THIS SECTION

- Overview | 689
- Configuration | 690
- Platform Support | 691
- Related Documentation | 691

Overview

IN THIS SECTION

- Benefits | 690

Equal-cost multipath (ECMP) is a network routing strategy that allows for traffic of the same session, or flow, to be transmitted across multiple paths of equal cost. A flow is traffic with the same source and destination. The ECMP process identifies routers that are legitimate equal-cost next hops toward the flow's destination. The device then uses load balancing to evenly distribute traffic across these multiple equal-cost next hops. ECMP is a mechanism that enables you (the network administrator) to load-balance traffic and increase bandwidth by fully utilizing otherwise unused bandwidth on links to the same destination.

You often use ECMP with BGP. Each BGP route can have multiple ECMP next hops. The BGP export policy determines whether to advertise the BGP route to these next hops. As the network administrator,

you can control the advertisement and withdrawal of BGP prefixes to and from these ECMP peers. The BGP export policy determines whether to advertise a BGP prefix based on the number of ECMP BGP peers the policy receives the prefix from.

You can configure the BGP export policy to withdraw a BGP route unless it receives the BGP route prefix from a minimum number of ECMP BGP peers. Requiring the BGP route to have multiple ECMP BGP peers creates better resiliency in case of link failures.

Benefits

- Improves resiliency of your network
- Prevents accidental overloading of links
- Assists with load balancing

Configuration

The BGP export policy compares the number of ECMP next hops for the BGP route against the value you configure with the `from nexthop-ecmp` statement at either of these hierarchies: `[edit policy-options policy-statement` *`policy-name`*`]` or `[edit policy-options policy-statement` *`policy-name`* `term` *`term-name`*`]`.

The options for this statement are:

- *`value`*: The exact number of ECMP gateways (1 through 512) required to meet the condition.
- `equal`: The number of gateways must be equal to the configured value.
- `greater-than`: The number of gateways must be greater than the configured value.
- `greater-than-equal`: The number of gateways must be greater than or equal to the configured value.
- `less-than`: The number of gateways must be less than the configured value.
- `less-than-equal`: The number of gateways must be less than or equal to the configured value.

1. Configure the BGP export policy to compare the number of ECMP next hops for the BGP route against the value you configure with the `from nexthop-ecmp` statement.

   In this example, the policy term `min-ecmp` finds a match when a route has less than two ECMP BGP peers.

   ```
   set policy-options policy-statement policy-name term min-ecmp from nexthop-ecmp less-than 2
   ```

2. Configure the BGP export policy to stop advertising BGP route prefixes if the number of ECMP next hops doesn't match the conditions you configured.

```
set policy-options policy-statement policy-name term min-ecmp then reject
set policy-options policy-statement policy-name term default then accept
```

3. Apply the policy to routes being exported from the routing table into BGP.

```
set protocols bgp group group-name export policy-name
```

4. Confirm that you have validated the value to be in line with the configured BGP ECMP peers in the policy.

```
show policy policy-name
```

5. Check whether the BGP route has been advertised to or withdrawn from the desired upstream BGP peer.

```
show route advertising-protocol bgp peer-advertised [detail]
```

Platform Support

See Feature Explorer for platform and release support.

Related Documentation

- Configuring Consistent Load Balancing for ECMP Groups

Understanding Entropy Label for BGP Labeled Unicast LSP

IN THIS SECTION

- What Is an Entropy Label? | 692
- Entropy Label for BGP Labeled Unicast | 693
- Supported and Unsupported Features | 694

What Is an Entropy Label?

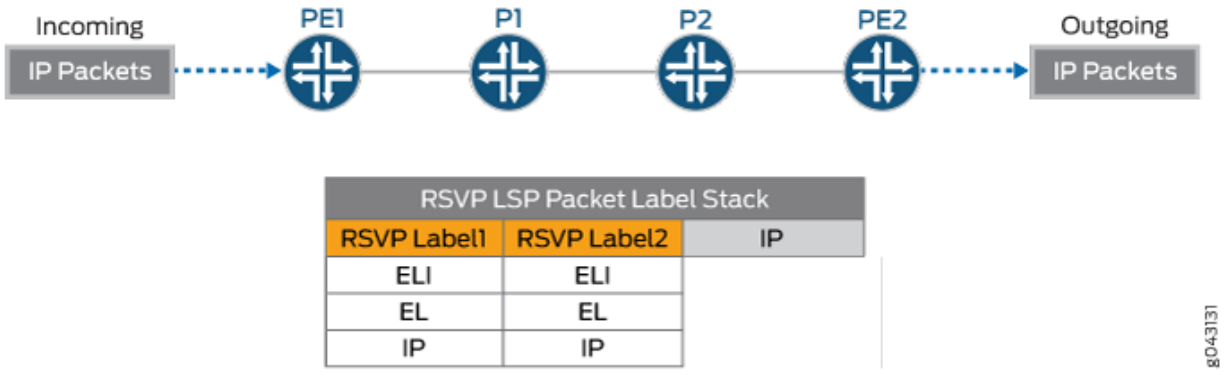
An entropy label is a special load-balancing label that enhances the router's ability to load-balance traffic across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs). The entropy label allows routers to efficiently load-balance traffic using just the label stack rather than deep packet inspection (DPI). DPI requires more of the router's processing power and is not a capability shared by all routers.

When an IP packet has multiple paths to reach its destination, Junos OS uses certain fields of the packet headers to hash the packet to a deterministic path. The source or destination addresses and port numbers of the packet are used to hash, in order to avoid packet reordering of a given flow. If a core label-switching router (LSR) is not capable of performing a DPI to identify the flow or can not do so at line rate, the label stack alone is used for ECMP hashing. This requires an entropy label, a special load-balancing label that can carry the flow information. The ingress LSR has more context and information about incoming packets than transit LSRs. Therefore, the ingress label edge router (LER) can inspect the flow information of a packet, map it to an entropy label, and insert it into the label stack. LSRs in the core simply use the entropy label as the key to hash the packet to the right path.

An entropy label can be any label value between 16 to 1048575 (regular 20-bit label range). Since this range overlaps with the existing regular label range, a special label called entropy label indicator (ELI) is inserted before the entropy label. ELI is a special label assigned by IANA with the value of 7.

Figure 50 on page 692 illustrates the entropy label in an RSVP label-switched path (LSP) packet label stack. The label stack consists of the entropy label indicator (ELI), the entropy label, and the IP packet.

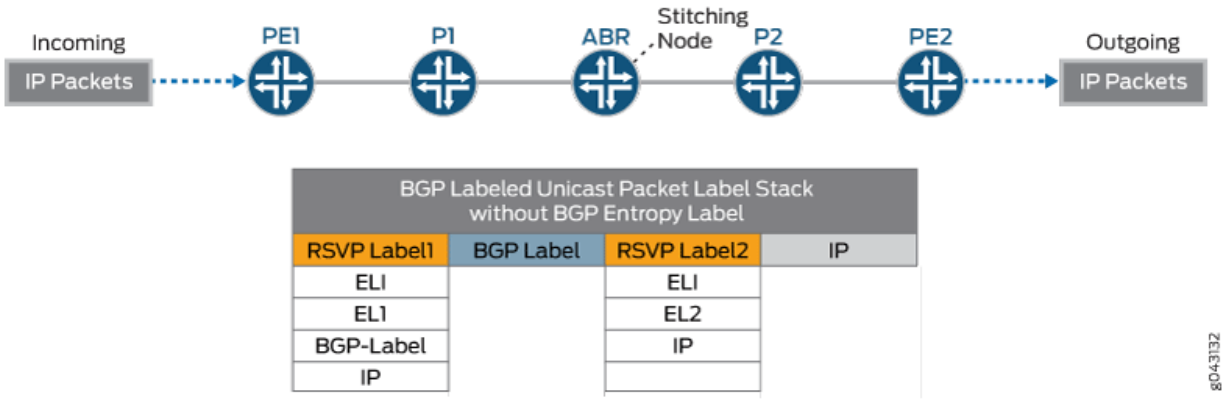
Figure 50: Entropy Label for RSVP LSP



Entropy Label for BGP Labeled Unicast

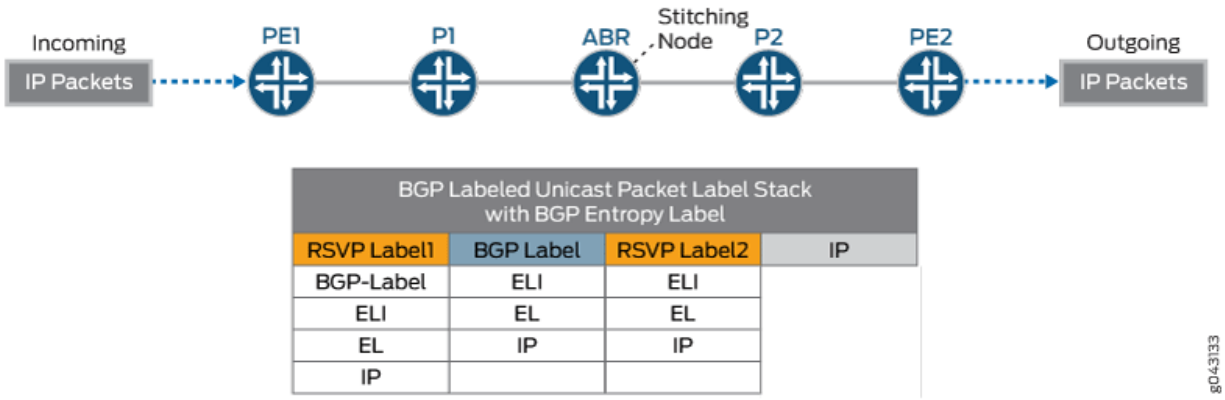
BGP labeled unicasts concatenate RSVP or LDP LSPs across multiple interior gateway protocol (IGP) areas or multiple autonomous systems (inter-AS LSPs). Inter-area BGP labeled unicast LSPs usually carry VPN and IP traffic when ingress PEs and egress PEs are in different IGP areas. When BGP labeled unicasts concatenate RSVP or LDP LSPs, Junos OS inserts the entropy labels at the BGP labeled unicast LSP ingress to achieve end-to-end entropy label load balancing. This is because RSVP or LDP entropy labels are usually popped at the penultimate hop node, together with the RSVP or LDP label, and there are no entropy labels at the stitching points, that is, the routers between two areas or two ASs. Therefore, in the absence of entropy labels, the router at the stitching point uses the BGP labels to forward packets. Figure 51 on page 693 illustrates the BGP labeled unicast packet label stack with the entropy label in an RSVP label stack. The RSVP label stack consists of the entropy label indicator (ELI), the entropy label, the BGP label, and the IP packet. The RSVP entropy labels are popped at the penultimate hop node.

Figure 51: Inter-Area BGP Labeled Unicast with RSVP Entropy Label



The BGP labeled unicast stitching node cannot use the entropy labels for load balancing unless the stitching node signals the entropy label capability at the BGP egress. If the BGP labeled unicast stitching node signals BGP entropy label capability (ELC) to the provider edge routers, the BGP labeled unicast LSP ingress is aware that the BGP labeled unicast LSP egress can handle entropy labels and inserts an entropy label indicator and entropy label underneath the BGP label. All of the LSRs are able to use the entropy label for load balancing. While BGP labeled unicast LSP might cross many routers in different areas and ASs, it is possible that some of the segments might support entropy labels while others might not. Figure 52 on page 694 illustrates the entropy label in the BGP label stack. The label stack at the stitching node consists of the ELI, the entropy label, and the IP packet.

Figure 52: Inter-Area BGP Labeled Unicast with BGP Entropy Label at Stitching Point



NOTE: To disable entropy label capability for BGP labeled unicast at the egress node, define a policy with the option `no-entropy-label-capability` at the `[edit policy-options policy-statement` *`policy-name`* `then]` hierarchy level.

```
[edit policy-options policy-statement policy-name then]
user@PE# no-entropy-label-capability
```

By default, routers that support entropy labels are configured with the *load-balance-label-capability* statement at the `[edit forwarding-options]` hierarchy level to signal the labels on a per-LSP basis. If the peer router is not equipped to handle load-balancing labels, you can prevent the signaling of entropy label capability by configuring the `no-load-balance-label-capability` statement at the `[edit forwarding-options]` hierarchy level.

```
[edit forwarding-options]
user@PE# no-load-balance-label-capability
```

By default, a BGP speaker uses the Entropy Label Capability (ELCv3) attribute defined within the IETF BGP Router Capability Attribute (RCA) for load balancing. It sends and receives only the ELCv3 attribute. If you need to use the ELCv2 attribute interoperable with the RCA draft, explicitly configure the `elc-v2-compatible` knob at the labeled-unicast entropy-label hierarchy. In such a scenario, both ELCv3 and ELCv2 are sent and received.

Supported and Unsupported Features

Junos OS supports an entropy label for BGP labeled unicast in the following scenarios:

- All the nodes of the LSPs have entropy label capability.
- Some of the nodes of the LSPs have entropy label capability.
- The LSPs tunnel through another carrier's VPN.
- Define an ingress policy to select a subset of BGP labeled unicast LSPs to insert an entropy label at ingress.
- Define an egress policy to disable entropy label capability advertisement.

Junos OS does not support the following features for an entropy label for BGP labeled unicast:

- When BGP labeled unicast LSPs are tunneling through another carrier's VPN, there is no true end-to-end entropy label because Junos OS does not insert an entropy label indicator or entropy label underneath VPN labels at the carrier-of-carriers network.
- Currently, Junos OS does not support IPv6 BGP labeled unicast LSPs with their own entropy labels. However, IPv6 BGP labeled unicast LSPs might use the entropy labels from the underlying RSVP, LDP, or BGP LSPs.

SEE ALSO

> *entropy-label*
>
> Example: Configuring an Entropy Label for a BGP Labeled Unicast LSP | **697**

Configuring an Entropy Label for a BGP Labeled Unicast LSP

Configure an entropy label for BGP labeled unicast LSP to achieve end-to-end entropy label load balancing. An entropy label is a special load-balancing label that can carry the flow information of the packets. BGP labeled unicasts generally concatenate RSVP or LDP LSPs across multiple IGP areas or multiple autonomous systems (ASs). RSVP or LDP entropy labels are popped at the penultimate hop node, together with the RSVP or LDP label. This feature enables the use of an entropy label at the stitching point, that is, the routers between two areas or ASs, to achieve end-to-end entropy label load balancing for BGP traffic. This feature enables the insertion of entropy labels at the BGP labeled unicast LSP ingress.

An entropy label can be any label value between 16 to 1048575 (regular 20-bit label range). Since this range overlaps with the existing regular label range, a special label called entropy label indicator (ELI) is inserted before the entropy label. ELI is a special label assigned by IANA with the value of 7.

Before you configure an entropy label for BGP labeled unicast, make sure you:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure LDP.
5. Configure RSVP.
6. Configure MPLS.

To configure an entropy label for BGP labeled unicast LSP:

1. On the ingress router, include the `entropy-label` statement at the `[edit protocols bgp family inet labeled-unicast]` hierarchy level to enable entropy label capability for BGP labeled unicast at a global level.

   You can also enable the use of an entropy label at a BGP group or a specific BGP neighbor level by including the `entropy-label` statement at the `[edit protocols bgp group` *`group name`* `family inet labeled-unicast]` or `[edit protocols bgp group` *`group name`* `neighbor` *`address`* `labeled-unicast]` hierarchy level.

   ```
   [edit protocols bgp family inet labeled-unicast]
   user@host# entropy-label
   ```

2. (Optional) Specify an additional policy to define the routes that have the entropy label capability.

   Apply the policy at the ingress router.

   ```
   [edit protocols bgp family inet labeled-unicast entropy-label]
   user@host# import policy-name;
   ```

3. (Optional) Include the option `no-next-hop-validation` if you do not want Junos OS to validate the next-hop field in the entropy label capability attribute against the route next hop.

   ```
   [edit protocols bgp family inet labeled-unicast entropy-label]
   user@host# no-next-hop-validation
   ```

4. (Optional) To explicitly disable advertising entropy label capability on the egress router, define a policy with the `no-entropy-label-capability` option for routes specified in the policy, and include the `no-entropy-label-capability` option in the specified policy at the `[edit policy-options policy statement` *`policy-name`* `then]` hierarchy level.

   ```
   [edit policy-options policy-statement policy-name then]
   user @ host# no-entropy-label-capability
   ```

SEE ALSO

entropy-label

Understanding Entropy Label for BGP Labeled Unicast LSP | **691**

Example: Configuring an Entropy Label for a BGP Labeled Unicast LSP

IN THIS SECTION

- Requirements | **697**
- Overview | **698**
- Configuration | **700**
- Verification | **714**

This example shows how to configure an entropy label for a BGP labeled unicast to achieve end-to-end load balancing using entropy labels. When an IP packet has multiple paths to reach its destination, Junos OS uses certain fields of the packet headers to hash the packet to a deterministic path. This requires an entropy label, a special load-balancing label that can carry the flow information. LSRs in the core simply use the entropy label as the key to hash the packet to the correct path. An entropy label can be any label value between 16 to 1048575 (regular 20-bit label range). Since this range overlaps with the existing regular label range, a special label called entropy label indicator (ELI) is inserted before the entropy label. ELI is a special label assigned by IANA with the value of 7.

BGP labeled unicasts generally concatenate RSVP or LDP LSPs across multiple IGP areas or multiple autonomous systems. RSVP or LDP entropy labels are popped at the penultimate hop node, together with the RSVP or LDP label. This feature enables the use of entropy labels at the stitching points to bridge the gap between the penultimate hop node and the stitching point, in order to achieve end-to-end entropy label load balancing for BGP traffic.

Requirements

This example uses the following hardware and software components:

- Seven MX Series routers with MPCs
- Junos OS Release 15.1 or later running on all the devices
  - Revalidated using Junos OS Relese 22.4

Before you configure an entropy label for BGP labeled unicast, make sure you:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure BGP.
4. Configure RSVP.
5. Configure MPLS.

Overview

IN THIS SECTION

- Topology | **699**

When BGP labeled unicasts concatenate RSVP or LDP LSPs across multiple IGP areas or multiple autonomous systems, RSVP or LDP entropy labels are popped at the penultimate hop node, together with the RSVP or LDP label. However, there are no entropy labels at the stitching points, that is, the routers between two areas. Therefore, the routers at the stitching points used the BGP labels to forward packets.

Beginning with Junos OS Release 15.1, you can configure an entropy label for BGP labeled unicast to achieve end-to-end entropy label load balancing. This feature enables the use of an entropy label at the stitching points in order to achieve end-to-end entropy label load balancing for BGP traffic. Junos OS allows the insertion of entropy labels at the BGP labeled unicast LSP ingress.

By default, routers that support entropy labels are configured with the `load-balance-label-capability` statement at the `[edit forwarding-options]` hierarchy level to signal the labels on a per-LSP basis. If the peer router is not equipped to handle load-balancing labels, you can prevent the signaling of entropy label capability by configuring the `no-load-balance-label-capability` at the `[edit forwarding-options]` hierarchy level.

```
[edit forwarding-options]
user@PE# no-load-balance-label-capability
```

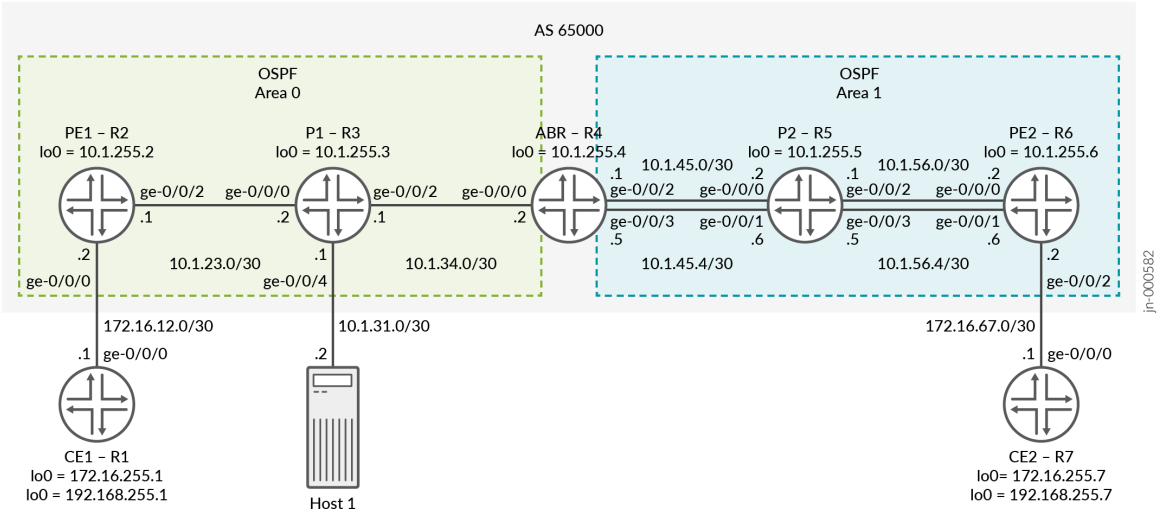
NOTE: You can explicitly disable advertising entropy label capability at egress for routes specified in the policy with the `no-entropy-label-capability` option at the `[edit policy-options policy-statement` *`policy name`* `then]` hierarchy level.

```
[edit policy-options policy-statement policy-name then]
user@PE# no-entropy-label-capability
```

Topology

In Figure 53 on page 699 , Router PE1 is the ingress router and Router PE2 is the egress router. Routers P1 and P2 are the transit routers. Router ABR is the area bridge router between Area 0 and Area 1. Two LSPs are configured on the ABR to PE2 for load balancing the traffic. Entropy label capability for BGP labeled unicast is enabled on the ingress Router PE1. Host 1 is connected to P1 for packet captures so that we can show the entropy label.

Figure 53: Configuring an Entropy Label for BGP Labeled Unicast



Configuration

IN THIS SECTION

- CLI Quick Configuration | 700
- Configuring Router PE1 | 705
- Configuring Router P1 | 708
- Configuring Router ABR | 710
- (Optional) Port-Mirroring Configuration | 712

CLI Quick Configuration

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

Router CE1

```
set interfaces ge-0/0/0 unit 0 family inet address 172.16.12.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.1/32 primary
set interfaces lo0 unit 0 family inet address 192.168.255.1/32
set routing-options router-id 172.16.255.1
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Router PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 172.16.12.2/30
set interfaces ge-0/0/2 unit 0 family inet address 10.1.23.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.255.2/32 primary
set interfaces lo0 unit 1 family inet address 10.1.255.22/32
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-instances VPN-l3vpn instance-type vrf
set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface lo0.1 passive
```

```
set routing-instances VPN-l3vpn protocols ospf export bgp-to-ospf
set routing-instances VPN-l3vpn interface ge-0/0/0.0
set routing-instances VPN-l3vpn interface lo0.1
set routing-instances VPN-l3vpn route-distinguisher 10.1.255.2:1
set routing-instances VPN-l3vpn vrf-target target:65000:1
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.1.255.2
set protocols bgp group ibgp family inet labeled-unicast entropy-label
set protocols bgp group ibgp neighbor 10.1.255.4 family inet labeled-unicast rib inet.3
set protocols bgp group ibgp neighbor 10.1.255.6 family inet-vpn unicast
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path pe1-abr to 10.1.255.4
set protocols mpls label-switched-path pe1-abr entropy-label
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols rsvp interface ge-0/0/2.0
set protocols rsvp interface lo0.0
```

Router P1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.23.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 10.1.34.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.255.3/32 primary
set routing-options router-id 10.1.255.3
set protocols mpls icmp-tunneling
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/2.0
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-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols rsvp interface ge-0/0/0.0
```

```
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/2.0
```

Router ABR

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.34.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 10.1.45.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 family inet address 10.1.45.5/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.255.4/32 primary
set forwarding-options hash-key family mpls label-1
set forwarding-options hash-key family mpls label-2
set forwarding-options hash-key family mpls label-3
set forwarding-options enhanced-hash-key family mpls no-payload
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement send-inet3-pe1 from route-filter 10.1.255.2/32 exact
set policy-options policy-statement send-inet3-pe1 then accept
set policy-options policy-statement send-inet3-pe2 from route-filter 10.1.255.6/32 exact
set policy-options policy-statement send-inet3-pe2 then accept
set routing-options router-id 10.1.255.4
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.1.255.4
set protocols bgp group ibgp family inet labeled-unicast rib inet.3
set protocols bgp group ibgp neighbor 10.1.255.2 export send-inet3-pe2
set protocols bgp group ibgp neighbor 10.1.255.6 export send-inet3-pe1
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path abr-pe1 to 10.1.255.2
set protocols mpls label-switched-path abr-pe1 entropy-label
set protocols mpls label-switched-path abr-pe2 to 10.1.255.6
set protocols mpls label-switched-path abr-pe2 entropy-label
set protocols mpls label-switched-path abr-pe2 primary to-r6-1
set protocols mpls label-switched-path abr-pe2-2 to 10.1.255.6
set protocols mpls label-switched-path abr-pe2-2 entropy-label
set protocols mpls label-switched-path abr-pe2-2 primary to-r6-2
set protocols mpls path to-r6-1 10.1.45.2 strict
set protocols mpls path to-r6-1 10.1.56.2 strict
set protocols mpls path to-r6-2 10.1.45.6 strict
set protocols mpls path to-r6-2 10.1.56.6 strict
```

```
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/0/3.0
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-0/0/0.0
set protocols ospf area 0.0.0.1 interface ge-0/0/2.0
set protocols ospf area 0.0.0.1 interface ge-0/0/3.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/2.0
set protocols rsvp interface ge-0/0/3.0
```

Router P2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.45.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 10.1.45.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 10.1.56.1/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 family inet address 10.1.56.5/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.255.5/32 primary
set forwarding-options hash-key family mpls label-1
set forwarding-options hash-key family mpls label-2
set forwarding-options hash-key family mpls label-3
set forwarding-options enhanced-hash-key family mpls no-payload
set policy-options policy-statement pplb then load-balance per-packet
set routing-options router-id 10.1.255.5
set routing-options forwarding-table export pplb
set protocols mpls icmp-tunneling
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface ge-0/0/3.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.1 interface lo0.0 passive
set protocols ospf area 0.0.0.1 interface ge-0/0/2.0
set protocols ospf area 0.0.0.1 interface ge-0/0/0.0
```

```
set protocols ospf area 0.0.0.1 interface ge-0/0/1.0
set protocols ospf area 0.0.0.1 interface ge-0/0/3.0
set protocols rsvp interface ge-0/0/2.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/3.0
```

Router PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.56.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 10.1.56.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 172.16.67.2/30
set interfaces lo0 unit 0 family inet address 10.1.255.6/32 primary
set interfaces lo0 unit 1 family inet address 10.1.255.66/32
set forwarding-options hash-key family mpls label-1
set forwarding-options hash-key family mpls label-2
set forwarding-options hash-key family mpls label-3
set forwarding-options enhanced-hash-key family mpls no-payload
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-instances VPN-l3vpn instance-type vrf
set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface lo0.1 passive
set routing-instances VPN-l3vpn protocols ospf export bgp-to-ospf
set routing-instances VPN-l3vpn interface ge-0/0/2.0
set routing-instances VPN-l3vpn interface lo0.1
set routing-instances VPN-l3vpn route-distinguisher 10.1.255.6:1
set routing-instances VPN-l3vpn vrf-target target:65000:1
set routing-options router-id 10.1.255.6
set routing-options autonomous-system 65000
set routing-options forwarding-table export pplb
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.1.255.6
set protocols bgp group ibgp family inet labeled-unicast entropy-label
set protocols bgp group ibgp neighbor 10.1.255.4 family inet labeled-unicast rib inet.3
set protocols bgp group ibgp neighbor 10.1.255.2 family inet-vpn unicast
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path pe2-abr to 10.1.255.4
```

```
set protocols mpls label-switched-path pe2-abr entropy-label
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface lo0.0
set protocols mpls interface ge-0/0/1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.1 interface ge-0/0/0.0
set protocols ospf area 0.0.0.1 interface lo0.0 passive
set protocols ospf area 0.0.0.1 interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface lo0.0
set protocols rsvp interface ge-0/0/1.0
```

Router CE2

```
set interfaces ge-0/0/0 unit 0 family inet address 172.16.67.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.7/32 primary
set interfaces lo0 unit 0 family inet address 192.168.255.7/32
set routing-options router-id 172.16.255.7
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Configuring Router PE1

Step-by-Step Procedure

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

To configure Router PE1:

NOTE: Repeat this procedure for Router PE2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the physical interfaces. Ensure to configure `family mpls` on the core facing interface.

```
[edit]
user@PE1# set interfaces ge-0/0/0 unit 0 family inet address 172.16.12.2/30
```

```
user@PE1# set interfaces ge-0/0/2 unit 0 family inet address 10.1.23.1/30
user@PE1# set interfaces ge-0/0/2 unit 0 family mpls
```

2. Configure the loopback interfaces. The secondary loopback is optional and is applied under the routing instance in a later step.

```
[edit]
user@PE1# set interfaces lo0 unit 0 family inet address 10.1.255.2/32 primary
user@PE1# set interfaces lo0 unit 1 family inet address 10.1.255.22/32
```

3. Configure the router ID and the autonomous system number.

```
[edit]
user@PE1# set routing-options router-id 10.1.255.2
user@PE1# set routing-options autonomous-system 65000
```

4. Configure the OSPF protocol.

```
[edit]
user@PE1# set protocols ospf traffic-engineering
user@PE1# set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

5. Configure the RSVP protocol.

```
[edit]
user@PE1# set protocols rsvp interface ge-0/0/2.0
user@PE1# set protocols rsvp interface lo0.0
```

6. Configure the MPLS protocol and an LSP towards the ABR. Include the `entropy-label` option to add the entropy label to the MPLS label stack.

```
[edit protocols]
user@PE1# set protocols mpls icmp-tunneling
user@PE1# set protocols mpls label-switched-path pe1-abr to 10.1.255.4
user@PE1# set protocols mpls label-switched-path pe1-abr entropy-label
```

```
user@PE1# set protocols mpls interface ge-0/0/2.0
user@PE1# set protocols mpls interface lo0.0
```

7. Configure IBGP using `family inet labeled-unicast` for the ABR peering and `family inet-vpn` for the PE2 peering. Enable entropy label capability for BGP labeled unicast.

```
[edit]
user@PE1# set protocols bgp group ibgp type internal
user@PE1# set protocols bgp group ibgp local-address 10.1.255.2
user@PE1# set protocols bgp group ibgp family inet labeled-unicast entropy-label
user@PE1# set protocols bgp group ibgp neighbor 10.1.255.4 family inet labeled-unicast rib
inet.3
user@PE1# set protocols bgp group ibgp neighbor 10.1.255.6 family inet-vpn unicast
```

8. Define a policy to export BGP VPN routes into OSPF. The policy is applied under OSPF in the routing instance.

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

9. Define a load balancing policy and apply it under the `routing-options forwarding-table`. PE1 only has one path in the example therefore this step is not needed, but for this example we are applying the same load balancing policy on all devices.

```
[edit]
user@PE1# set policy-options policy-statement pplb then load-balance per-packet
user@PE1# set routing-options forwarding-table export pplb
```

10. Configure the Layer 3 VPN routing instance.

```
[edit]
user@PE1# set routing-instances VPN-l3vpn instance-type vrf
```

11. Assign the interfaces to the routing instance.

```
[edit]
user@PE1# set routing-instances VPN-l3vpn interface ge-0/0/0.0
user@PE1# set routing-instances VPN-l3vpn interface lo0.1
```

12. Configure the route distinguisher for the routing instance.

```
[edit]
user@PE1# set routing-instances VPN-l3vpn route-distinguisher 10.1.255.2:1
```

13. Configure a VPN routing and forwarding (VRF) target for the routing instance.

```
[edit]
user@PE1# set routing-instances VPN-l3vpn vrf-target target:65000:1
```

14. Configure the protocol OSPF under the routing instance and apply the previously configured `bgp-to-ospf` policy.

```
[edit]
user@PE1# set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface ge-0/0/0.0
user@PE1# set routing-instances VPN-l3vpn protocols ospf area 0.0.0.0 interface lo0.1
passive
user@PE1# set routing-instances VPN-l3vpn protocols ospf export bgp-to-ospf
```

Configuring Router P1

Step-by-Step Procedure

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

To configure Router P1:

NOTE: Repeat this procedure for Router P2 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the physical interfaces.

```
[edit]
user@P1# set interfaces ge-0/0/0 unit 0 family inet address 10.1.23.2/30
user@P1# set interfaces ge-0/0/0 unit 0 family mpls
user@P1# set interfaces ge-0/0/2 unit 0 family inet address 10.1.34.1/30
user@P1# set interfaces ge-0/0/2 unit 0 family mpls
```

2. Configure the loopback interface.

```
[edit]
user@P1# set interfaces lo0 unit 0 family inet address 10.1.255.3/32 primary
```

3. Configure the router ID.

```
[edit]
user@P1# set routing-options router-id 10.1.255.3
```

4. Configure the OSPF protocol.

```
[edit]
user@P1# set protocols ospf traffic-engineering
user@P1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P1# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
```

5. Configure the RSVP protocol.

```
[edit]
user@P1# set protocols rsvp interface ge-0/0/0.0
user@P1# set protocols rsvp interface lo0.0
user@P1# set protocols rsvp interface ge-0/0/2.0
```

6. Configure the MPLS protocol.

```
[edit]
user@P1# set protocols mpls icmp-tunneling
```

```
user@P1# set protocols mpls interface ge-0/0/0.0
user@P1# set protocols mpls interface lo0.0
user@P1# set protocols mpls interface ge-0/0/2.0
```

Configuring Router ABR

Step-by-Step Procedure

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

To configure Router ABR:

1. Configure the physical interfaces.

```
[edit]
user@ABR# set interfaces ge-0/0/0 unit 0 family inet address 10.1.34.2/30
user@ABR# set interfaces ge-0/0/0 unit 0 family mpls
user@ABR# set interfaces ge-0/0/2 unit 0 family inet address 10.1.45.1/30
user@ABR# set interfaces ge-0/0/2 unit 0 family mpls
user@ABR# set interfaces ge-0/0/3 unit 0 family inet address 10.1.45.5/30
user@ABR# set interfaces ge-0/0/3 unit 0 family mpls
```

2. Configure the loopback interface.

```
[edit]
user@ABR# set interfaces lo0 unit 0 family inet address 10.1.255.4/32 primary
```

3. Configure MPLS labels that the router uses for hashing the packets to its destination for load balancing.

```
[edit]
user@ABR# set forwarding-options hash-key family mpls label-1
user@ABR# set forwarding-options hash-key family mpls label-2
user@ABR# set forwarding-options hash-key family mpls label-3
user@ABR# set forwarding-options enhanced-hash-key family mpls no-payload
```

4. Configure the router ID and the autonomous system number.

```
[edit]
user@ABR# set routing-options router-id 10.1.255.4
user@ABR# set routing-options autonomous-system 65000
```

5. Configure the OSPF protocol.

```
[edit]
user@ABR# set protocols ospf traffic-engineering
user@ABR# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@ABR# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
user@ABR# set protocols ospf area 0.0.0.1 interface ge-0/0/2.0
user@ABR# set protocols ospf area 0.0.0.1 interface ge-0/0/3.0
```

6. Configure the RSVP protocol.

```
[edit]
user@ABR# set protocols rsvp interface lo0.0
user@ABR# set protocols rsvp interface ge-0/0/0.0
user@ABR# set protocols rsvp interface ge-0/0/2.0
user@ABR# set protocols rsvp interface ge-0/0/3.0
```

7. Configure the MPLS protocol and specify the LSPs towards PE1 and PE2. Two LSPs are created towards PE2 for the purpose of load balancing traffic to show different LSPs and interfaces are used.

```
[edit]
user@ABR# set protocols mpls icmp-tunneling
user@ABR# set protocols mpls label-switched-path abr-pe1 to 10.1.255.2
user@ABR# set protocols mpls label-switched-path abr-pe1 entropy-label
user@ABR# set protocols mpls label-switched-path abr-pe2 to 10.1.255.6
user@ABR# set protocols mpls label-switched-path abr-pe2 entropy-label
user@ABR# set protocols mpls label-switched-path abr-pe2 primary to-r6-1
user@ABR# set protocols mpls label-switched-path abr-pe2-2 to 10.1.255.6
user@ABR# set protocols mpls label-switched-path abr-pe2-2 entropy-label
user@ABR# set protocols mpls label-switched-path abr-pe2-2 primary to-r6-2
user@ABR# set protocols mpls path to-r6-1 10.1.45.2 strict
user@ABR# set protocols mpls path to-r6-1 10.1.56.2 strict
```

```
user@ABR# set protocols mpls path to-r6-2 10.1.45.6 strict
user@ABR# set protocols mpls path to-r6-2 10.1.56.6 strict
user@ABR# set protocols mpls interface lo0.0
user@ABR# set protocols mpls interface ge-0/0/0.0
user@ABR# set protocols mpls interface ge-0/0/2.0
user@ABR# set protocols mpls interface ge-0/0/3.0
```

8. Configure IBGP to both PE1 and PE2 using `family inet labeled-unicast`. Apply the policy to advertise the inet.3 loopback route from both PE1 and PE2. We show the policy in the next step.

```
[edit]
user@ABR# set protocols bgp group ibgp type internal
user@ABR# set protocols bgp group ibgp local-address 10.1.255.4
user@ABR# set protocols bgp group ibgp family inet labeled-unicast rib inet.3
user@ABR# set protocols bgp group ibgp neighbor 10.1.255.2 export send-inet3-pe2
user@ABR# set protocols bgp group ibgp neighbor 10.1.255.6 export send-inet3-pe1
```

9. Define a policy to match on the loopback addresses for PE1 and PE2.

```
[edit]
user@ABR# set policy-options policy-statement send-inet3-pe1 from route-filter
10.1.255.2/32 exact
user@ABR# set policy-options policy-statement send-inet3-pe1 then accept
user@ABR# set policy-options policy-statement send-inet3-pe2 from route-filter
10.1.255.6/32 exact
user@ABR# set policy-options policy-statement send-inet3-pe2 then accept
```

10. Define a policy for load balancing and apply it under the `routing-options forwarding-table`.

```
[edit]
user@ABR# set policy-options policy-statement pplb then load-balance per-packet
user@ABR# set routing-options forwarding-table export pplb
```

(Optional) Port-Mirroring Configuration

To see the entropy label that is applied you can capture the traffic. In this example a filter is applied on the PE1 facing interface on P1 to capture the CE1 to CE2 traffic. The traffic is sent to Host 1 for viewing. There are different ways to capture traffic than what we use in this example. For more information see No Link Title.

Step-by-Step Procedure

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

To configure Router P1:

1. Configure the interfaces. In this example we are putting the interface connected to Host1 in a bridge domain and creating an IRB interface for verifying connectivity to Host1.

```
[edit]
user@P1# set interfaces ge-0/0/4 unit 0 family bridge interface-mode access
user@P1# set interfaces ge-0/0/4 unit 0 family bridge vlan-id 100
user@P1# set interfaces irb unit 0 family inet address 10.1.31.1/30
```

2. Configure the bridge domain.

```
[edit]
user@P1# set bridge-domains v100 vlan-id 100
user@P1# set bridge-domains v100 routing-interface irb.0
```

3. Configure a filter to capture the traffic. For this example we are capturing all traffic.

```
[edit]
user@P1# set firewall family any filter test term 1 then count test
user@P1# set firewall family any filter test term 1 then port-mirror
user@P1# set firewall family any filter test term 1 then accept
```

4. Apply the filter to the PE1 facing interface.

```
[edit]
user@P1# set interfaces ge-0/0/0 unit 0 filter input test
```

5. Configure the port mirroring options. For this example we are mirroring all traffic and sending it to Host1 connected to interface ge-0/0/4.

```
[edit]
user@P1# set forwarding-options port-mirroring input rate 1
user@P1# set forwarding-options port-mirroring family any output interface ge-0/0/4.0
```

Verification

IN THIS SECTION

- Verifying That the Entropy Label Capability Is Being Advertised | **714**
- Verifying That Router PE1 Receives the Entropy Label Advertisement | **715**
- Verifying ECMP at the ABR to PE2 | **717**
- Show Routes to CE2 on PE1 | **718**
- Ping CE2 from CE1 | **720**
- Verify Load Balancing | **721**
- Verify the Entropy Label | **721**

Confirm that the configuration is working properly.

Verifying That the Entropy Label Capability Is Being Advertised

Purpose

Verify that the entropy label capability path attribute is being advertised from the ABR to PE1 for the route to PE2.

Action

From operational mode, run the **show route advertising-protocol bgp 10.1.255.2 detail** command on Router ABR.

```
user@ABR> show route advertising-protocol bgp 10.1.255.2 detail

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
* 10.1.255.6/32 (1 entry, 1 announced)
 BGP group ibgp type Internal
     Route Label: 299952
     Nexthop: Self
     Flags: Nexthop Change
     MED: 2
     Localpref: 4294967294
     AS path: [65000] I
     Entropy label capable
```

Meaning

The output shows that the host PE2 with the IP address of 10.1.255.6 has the entropy label capability and the route label that is used. The host is advertising the entropy label capability to its BGP neighbors.

Verifying That Router PE1 Receives the Entropy Label Advertisement

Purpose

Verify that Router PE1 receives the entropy label advertisement for Router PE2.

Action

From operational mode, run the **show route protocol bgp 10.1.255.6 extensive** command on Router PE1.

```
user@PE1> show route protocol bgp 10.1.255.6 extensive

inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.1.255.6/32 (1 entry, 1 announced)
        *BGP    Preference: 170/1
                Next hop type: Indirect, Next hop index: 0
                Address: 0x7b3ffd4
                Next-hop reference count: 2, key opaque handle: 0x0, non-key opaque handle: 0x0
                Source: 10.1.255.4
                Next hop type: Router, Next hop index: 0
                Next hop: 10.1.23.2 via ge-0/0/2.0, selected
                Label-switched-path pe1-abr
```

```
                Label operation: Push 299952, Push 299808(top)
                Label TTL action: prop-ttl, prop-ttl(top)
                Load balance label: Label 299952: Entropy label; Label 299808: None;
                Label element ptr: 0x93d6bf8
                Label parent element ptr: 0x93d6c20
                Label element references: 3
                Label element child references: 2
                Label element lsp id: 0
                Session Id: 0
                Protocol next hop: 10.1.255.4
                Label operation: Push 299952
                Label TTL action: prop-ttl
                Load balance label: Label 299952: Entropy label;
                Indirect next hop: 0x758c05c - INH Session ID: 0
                State: <Active Int Ext>
                Local AS: 65000 Peer AS: 65000
                Age: 1:33:11    Metric: 2       Metric2: 2
                Validation State: unverified
                Task: BGP_65000.10.1.255.4
                Announcement bits (2): 3-Resolve tree 1 4-Resolve_IGP_FRR task
                AS path: I
                Accepted
                Route Label: 299952
                Localpref: 4294967294
                Router ID: 10.1.255.4
                Session-IDs associated:
                Session-id: 324 Version: 3
                Thread: junos-main
                Indirect next hops: 1
                        Protocol next hop: 10.1.255.4 Metric: 2 ResolvState: Resolved
                        Label operation: Push 299952
                        Label TTL action: prop-ttl
                        Load balance label: Label 299952: Entropy label;
                        Indirect next hop: 0x758c05c - INH Session ID: 0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.1.23.2 via ge-0/0/2.0
                                Session Id: 0
                                10.1.255.4/32 Originating RIB: inet.3
                                  Metric: 2 Node path count: 1
                                  Forwarding nexthops: 1
                                        Next hop type: Router
```

```
                                        Next hop: 10.1.23.2 via ge-0/0/2.0
                                        Session Id: 0
```

Meaning

Router PE1 receives the entropy label capability advertisement from its BGP neighbor.

Verifying ECMP at the ABR to PE2

Purpose

Verify equal-cost multipath (ECMP) to PE2.

Action

From operational mode, run the **show route table mpls.0** and **show route forwarding-table label <label>**commands on Router ABR.

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

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 2w1d 23:02:11, metric 1
                       Receive
1                  *[MPLS/0] 2w1d 23:02:11, metric 1
                       Receive
2                  *[MPLS/0] 2w1d 23:02:11, metric 1
                       Receive
13                 *[MPLS/0] 2w1d 23:02:11, metric 1
                       Receive
299936             *[VPN/170] 2d 21:47:02
                    >  to 10.1.34.1 via ge-0/0/0.0, label-switched-path abr-pe1
299952             *[VPN/170] 2d 21:47:02
                    >  to 10.1.45.2 via ge-0/0/2.0, label-switched-path abr-pe2
                       to 10.1.45.6 via ge-0/0/3.0, label-switched-path abr-pe2-2

ruser@ABR> show route forwarding-table label 299952
Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
```

```
299952             user     0                    ulst  1048575     2
                              10.1.45.2         Swap 299824      516     2 ge-0/0/2.0
                              10.1.45.6         Swap 299840      572     2 ge-0/0/3.0
...
```

Meaning

The output shows an ECMP for the label used for the BGP labeled unicast route.

Show Routes to CE2 on PE1

Purpose

Verify the routes to CE2.

Action

From operational mode, run the **show route table VPN-l3vpn.inet.0 172.16.255.7 extensive** and **show route table VPN-l3vpn.inet.0 192.168.255.7 extensive**commands on Router PE1.

```
user@PE1> show route table VPN-l3vpn.inet.0 172.16.255.7 extensive

VPN-l3vpn.inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
172.16.255.7/32 (1 entry, 1 announced)
TSI:
OSPF area : 0.0.0.0, LSA ID : 172.16.255.7, LSA type : Summary
KRT in-kernel 172.16.255.7/32 -> {indirect(1048574)}
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.1.255.6:1
                Next hop type: Indirect, Next hop index: 0
                Address: 0x7b40434
                Next-hop reference count: 9, key opaque handle: 0x0, non-key opaque handle: 0x0
                Source: 10.1.255.6
                Next hop type: Router, Next hop index: 515
                Next hop: 10.1.23.2 via ge-0/0/2.0, selected
                Label-switched-path pe1-abr
                Label operation: Push 299824, Push 299952, Push 299808(top)
                Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
                Load balance label: Label 299824: None; Label 299952: Entropy label; Label
299808: None;
                Label element ptr: 0x93d6c98
```

```
                Label parent element ptr: 0x93d6bf8
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 140
                Protocol next hop: 10.1.255.6
                Label operation: Push 299824
                Label TTL action: prop-ttl
                Load balance label: Label 299824: None;
...

user@PE1> show route table VPN-l3vpn.inet.0 192.168.255.7 extensive
VPN-l3vpn.inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
192.168.255.7/32 (1 entry, 1 announced)
TSI:
OSPF area : 0.0.0.0, LSA ID : 192.168.255.7, LSA type : Summary
KRT in-kernel 192.168.255.7/32 -> {indirect(1048574)}
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.1.255.6:1
                Next hop type: Indirect, Next hop index: 0
                Address: 0x7b40434
                Next-hop reference count: 9, key opaque handle: 0x0, non-key opaque handle: 0x0
                Source: 10.1.255.6
                Next hop type: Router, Next hop index: 515
                Next hop: 10.1.23.2 via ge-0/0/2.0, selected
                Label-switched-path pe1-abr
                Label operation: Push 299824, Push 299952, Push 299808(top)
                Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
                Load balance label: Label 299824: None; Label 299952: Entropy label; Label
299808: None;
                Label element ptr: 0x93d6c98
                Label parent element ptr: 0x93d6bf8
                Label element references: 1
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 140
                Protocol next hop: 10.1.255.6
                Label operation: Push 299824
                Label TTL action: prop-ttl
                Load balance label: Label 299824: None;
...
```

Meaning

The output shows the same labels are used for both routes.

Ping CE2 from CE1

Purpose

Verify connectivity and to use for verifying load balancing.

Action

From operational mode, run the **ping 172.16.255.7 source 172.16.12.1 rapid count 100** and **ping 192.168.255.7 source 192.168.255.1 rapid count 200**commands on Router PE1.

```
user@CE1> ping 172.16.255.7 source 172.16.12.1 rapid count 100
PING 172.16.255.7 (172.16.255.7): 56 data bytes
!!!
```

Verify Load Balancing

Purpose

Verify load balancing.

Action

From operational mode, run the **show mpls lsp ingress statistics** command on the ABR.

```
user@ABR> show mpls lsp ingress statistics
Ingress LSP: 3 sessions
To              From            State     Packets            Bytes LSPname
10.1.255.2      10.1.255.4      Up            300            30000 abr-pe1
10.1.255.6      10.1.255.4      Up            200            20000 abr-pe2
10.1.255.6      10.1.255.4      Up            100            10000 abr-pe2-2
Total 3 displayed, Up 3, Down 0
```

Meaning

The output shows the first ping from the previous command used LSP **abr-pe2-2** and the second ping used LSP **abr-pe2**.

Verify the Entropy Label

Purpose

Verify the entropy label is different between the pings that were used.

Action

On Host 1, run the **tcpdump -i eth1 -n**.

```
user@Host1# tcpdump -i eth1 -n
...
13:42:31.993274 MPLS (label 299808, exp 0, ttl 63) (label 299952, exp 0, ttl 63) (label 7, exp
0, ttl 63) (label 1012776, exp 0, ttl 0)
(label 299824, exp 0, [S], ttl 63) IP 172.16.12.1 > 172.16.255.7: ICMP echo request, id 32813,
seq 9, length 64
```

```
...
13:43:19.570260 MPLS (label 299808, exp 0, ttl 63) (label 299952, exp 0, ttl 63) (label 7, exp
0, ttl 63) (label 691092, exp 0, ttl 0)
(label 299824, exp 0, [S], ttl 63) IP 192.168.255.1 > 192.168.255.7: ICMP echo request, id
46381, seq 9, length 64
```

Meaning

The output shows the different value for the entropy label for the two different ping commands.

Use Case for BGP Prefix Independent Convergence for Inet, Inet6, or Labeled Unicast

In the instance of a router failure, a BGP network can take from a few seconds to minutes to recover, depending on parameters such as the size of the network or router performance. When the BGP Prefix Independent Convergence (PIC) feature is enabled on a router, BGP installs to the Packet Forwarding Engine the second best path in addition to the calculated best path to a destination. The router uses this backup path when an egress router fails in a network and drastically reduces the outage time. You can enable this feature to reduce the network downtime if the egress router fails.

When reachability to an egress router in a network fails, the IGP detects this outage, and the link state propagates this information throughout the network and advertises the BGP next hop for that prefix as unreachable. BGP reevaluates alternative paths and if an alternative path is available, reinstalls this alternate next hop into the Packet Forwarding Engine. This kind of egress failure usually impacts multiple prefixes at the same time, and BGP has to update all these prefixes one at a time. On the ingress routers, the IGP completes the shortest path first (SPF) and updates the next hops. Junos OS then determines the prefixes that have become unreachable and signals to the protocol that these need to be updated. BGP gets the notification and updates the next hop for every prefix that is now invalid. This process could impact the connectivity and could take a few minutes to recover from the outage. BGP PIC can reduce this down time as the backup path is already installed in the Packet Forwarding Engine.

Beginning with Junos OS Release 15.1, the BGP PIC feature, which was initially supported for Layer 3 VPN routers, is extended to BGP with multiple routes in the global tables such as inet and inet6 unicast, and inet and inet6 labeled unicast. On a BGP PIC enabled router, Junos OS installs the backup path for the indirect next hop on the Routing Engine and also provides this route to the Packet Forwarding Engine and IGP. When an IGP loses reachability to a prefix with one or more routes, it signals to the Routing Engine with a single message prior to updating the routing tables. The Routing Engine signals to the Packet Forwarding Engine that an indirect next hop has failed, and traffic must be rerouted using the backup path. Routing to the impacted destination prefix continues using the backup path even before

BGP starts recalculating the new next hops for the BGP prefixes. The router uses this backup path to reduce traffic loss until the global convergence through the BGP is resolved.

The time at which the outage occurs to the time until the loss of reachability is signaled actually depends on the failure detection time of the nearest router and the IGP convergence time. Once the local router detects the outage, the route convergence without the BGP PIC feature enabled depends heavily on the number of prefixes affected and the performance of the router due to recalculation of each affected prefix. However, with the BGP PIC feature enabled, even before BGP recalculates the best path for those affected prefixes, the Routing Engine signals the data plane to switch to the standby next best path. Hence traffic loss is minimum. The new routes are calculated even while the traffic is being forwarded, and these new routes are pushed down to the data plane. Therefore, the number of BGP prefixes affected does not impact the time taken from the time traffic outage occurs to the point of time at which BGP signals the loss of reachability.

SEE ALSO

> *Configuring BGP PIC Edge for MPLS Layer 3 VPNs*
>
> Example: Configuring BGP Prefix Independent Convergence for Inet | **728**
>
> *Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs*

Configuring BGP Prefix Independent Convergence for Inet

On a BGP Prefix Independent Convergence (PIC) enabled router, Junos OS installs the backup path for the indirect next hop on the Routing Engine and also provides this route to the Packet Forwarding Engine and IGP. When an IGP loses reachability to a prefix with one or more routes, it signals to the Routing Engine with a single message prior to updating the routing tables. The Routing Engine signals to the Packet Forwarding Engine that an indirect next hop has failed, and traffic must be rerouted using the backup path. Routing to the impacted destination prefix continues using the backup path even before BGP starts recalculating the new next hops for the BGP prefixes. The router uses this backup path to reduce traffic loss until the global convergence through the BGP is resolved. The BGP PIC feature, which was initially supported for Layer 3 VPN routers, is extended to BGP with multiple routes in the global tables such as inet and inet6 unicast, and inet and inet6 labeled unicast.

Before you begin:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and LDP.

4. Configure BGP.

NOTE: The BGP PIC feature is supported only on routers with MPC interfaces.

BEST PRACTICE: On routers with Modular Port Concentrators (MPCs), enable enhanced IP network services as shown here:

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

To configure BGP PIC for inet:

1. Enable BGP PIC for inet.

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

NOTE: The BGP PIC edge feature is supported only on 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.

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

4. Verify that BGP PIC is working.

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

```
user@host> show route 20.1.1.1 extensive
inet.0: 236941 destinations, 630411 routes (236940 active, 0 holddown, 1 hidden)
20.1.1.1/32 (3 entries, 2 announced)
        State: <CalcForwarding>
```

```
TSI:
KRT in-kernel 20.1.1.1/32 -> {indirect(1048574), indirect(1048575)}
        @BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xafd09d0
                Next-hop reference count: 236886
                Source: 10.255.183.55
                Next hop type: Router, Next hop index: 623
                Next hop: 100.0.1.2 via ge-2/1/2.0, selected
                Session Id: 0x140
                Protocol next hop: 10.255.183.55
                Indirect next hop: 0xab3b980 1048574 INH Session ID: 0x144
                State: <Active Int Ext ProtectionPath ProtectionCand>
                Local AS:   100 Peer AS:   100
                Age: 1:11       Metric2: 2
                Validation State: unverified
                Task: BGP_100.10.255.183.55
                Announcement bits (1): 6-Resolve tree 2
                AS path: 200 400 I
                Accepted MultipathUnequal
                Localpref: 100
                Router ID: 10.255.183.55
                Indirect next hops: 1
                                                Protocol next hop: 10.255.183.55 Metric:
2
                                                 Indirect next hop: 0xab3b980 1048574 INH
Session ID: 0x144
                                                Indirect path forwarding next hops:
1
                                                                Next hop type:
Router
                                                                Next hop: 100.0.1.2 via
ge-2/1/2.0
                                                                Session Id:
0x140
                                                10.255.183.55/32 Originating RIB:
inet.0
                                                    Metric: 2                       Node
path count: 1
                                                    Forwarding nexthops:
1
                                                                Nexthop: 100.0.1.2 via
ge-2/1/2.0
```

```
      BGP    Preference: 170/-101
                        Next hop type: Indirect, Next hop index: 0
                        Address: 0xafd0970
                        Next-hop reference count: 196735
                        Source: 10.255.183.56
                        Next hop type: Router, Next hop index: 624
                        Next hop: 100.0.2.2 via ge-2/0/9.0, selected
                        Session Id: 0x141
                          Protocol next hop: 10.255.183.56
                          Indirect next hop: 0xab3c240 1048575 INH Session ID:
0x145
                          State: <NotBest Int Ext ProtectionCand>
                          Inactive reason: Not Best in its group - IGP
metric
                          Local AS:   100 Peer AS:   100
                          Age: 1:05       Metric2: 1001
                          Validation State: unverified
                          Task: BGP_100.10.255.183.56
                          AS path: 200 400 I
                          Accepted
                          Localpref: 100
                          Router ID: 10.255.183.56
                          Indirect next hops: 1
                                        Protocol next hop: 10.255.183.56 Metric:
1001
                                        Indirect next hop: 0xab3c240 1048575 INH Session
ID: 0x145
                                        Indirect path forwarding next hops:
1
                                                        Next hop type:
Router
                                                        Next hop: 100.0.2.2 via
ge-2/0/9.0
                                                        Session Id:
0x141
                                        10.255.183.56/32 Originating RIB:
inet.0
                                            Metric: 1001                    Node path
count: 1
                                            Forwarding nexthops:
1
                                                        Nexthop: 100.0.2.2 via
ge-2/0/9.0
```

```
#Multipath Preference: 255
                                    Next hop type: Indirect, Next hop index:
0
                                    Address: 0xd330f90
                                    Next-hop reference count: 304062
                                    Next hop type: Router, Next hop index:
623
                                    Next hop: 100.0.1.2 via ge-2/1/2.0,
selected
                                    Session Id: 0x140
                                    Next hop type: Router, Next hop index:
624
                                    Next hop: 100.0.2.2 via ge-2/0/9.0
                                    Session Id: 0x141
                                    Protocol next hop: 10.255.183.55
                                    Indirect next hop: 0xab3b980 1048574 INH Session ID:
0x144 Weight 0x1
                                    Protocol next hop: 10.255.183.56
                                    Indirect next hop: 0xab3c240 1048575 INH Session ID:
0x145 Weight 0x4000
                                    State:   <ForwardinOnly Int Ext>
                                    Inactive reason: Forwarding use only
                                    Local AS:   100
                                    Age: 1:05       Metric2: 2
                                    Validation State: unverified
                                    Task: RT
                                    Announcement bits (1): 0-KRT
                                    AS path: 200 400 I

user@host> show route forwarding-table destination 20.1.1.1 extensive
  Routing table: default.inet [Index 0]
Internet:

Destination:  20.1.1.1/32
     Route type: user
     Route reference: 0                  Route interface-index: 0
     Multicast RPF nh index: 0
     Flags: sent to PFE
     Next-hop type: unilist              Index: 1048576  Reference: 7401
     Next-hop type: indirect             Index: 1048574  Reference:
2                                        Weight: 0x1
     Nexthop: 100.0.1.2
     Next-hop type: unicast              Index: 623      Reference: 8
```

```
     Next-hop interface: ge-2/1/2.0    Weight: 0x1
     Next-hop type: indirect             Index: 1048575  Reference:
2                                        Weight: 0x4000
     Nexthop: 100.0.2.2
     Next-hop type: unicast              Index: 624      Reference: 8
     Next-hop interface: ge-2/0/9.0    Weight: 0x4000
```

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.

SEE ALSO

Use Case for BGP Prefix Independent Convergence for Inet, Inet6, or Labeled Unicast | 722

Example: Configuring BGP Prefix Independent Convergence for Inet

IN THIS SECTION

- Requirements | 729
- Overview | 729
- Configuration | 730
- Verification | 743

This example shows how to configure BGP PIC for inet. In the instance of a router failure, a BGP network can take from a few seconds to minutes to recover, depending on parameters such as the size of the network or router performance. When the BGP Prefix Independent Convergence (PIC) feature is enabled on a router, BGP with multiple routes in the global tables, such as inet and inet6 unicast, and inet and inet6 labeled unicast, installs to the Packet Forwarding Engine the second best path in addition to the calculated best path to a destination. The router uses this backup path when an egress router fails in a network and drastically reduces the outage time.

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 router with MPCs to configure the BGP PIC feature
- Seven routers that can be a combination of M Series, MX Series, T Series, or PTX Series routers
- Junos OS Release 15.1 or later on the device with BGP PIC configured

Overview

IN THIS SECTION

- Topology | **729**

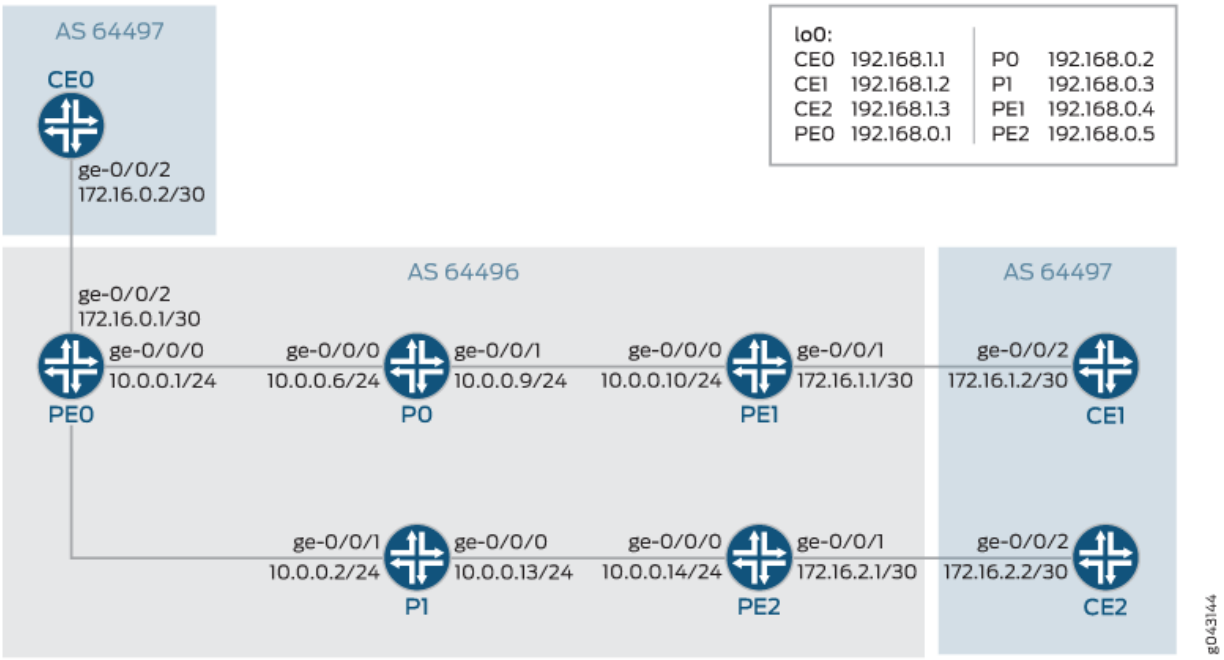
Beginning with Junos OS Release 15.1, BGP PIC, which was initially supported for Layer 3 VPN routers, is extended to BGP with multiple routes in the global tables such as inet and inet6 unicast, and inet and inet6 labeled unicast. BGP installs to the Packet Forwarding Engine the second best path in addition to the calculated best path to a destination. When an IGP loses reachability to a prefix, the router uses this backup path to reduce traffic loss until the global convergence through the BGP is resolved, thereby reducing the outage duration.

NOTE: The BGP PIC feature is supported only on routers with MPCs.

Topology

This example shows three customer edge (CE) routers, Device CE0, CE1, and CE2. Routers PE0, PE1, and PE2 are the provider edge (PE) routers. Router P0 and P1 are the provider core routers. BGP PIC is configured on Router PE0. For testing, the address 192.168.1.5 is added as a second loopback interface address on Device CE1. The address is announced to Routers PE1 and PE2 and is relayed by the internal BGP (IBGP) to Router PE0. On Router PE0, there are two paths to the 192.168.1.5 network. These are the primary path and a backup path. Figure 54 on page 730 shows the sample network.

Figure 54: Configuring BGP PIC for Inet



Configuration

IN THIS SECTION

- CLI Quick Configuration | **730**
- Configuring Device PE0 | **736**
- Results | **739**

CLI Quick Configuration

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

Router PE0

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 description PE0->P0
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.5/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::1/32
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description PE0->P1
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.1/24
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::2/32
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set interfaces ge-0/0/2 unit 0 description PE0->CE0
set interfaces ge-0/0/2 unit 0 family inet address 172.16.0.1/30
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8::10/32
set interfaces ge-0/0/2 unit 0 family mpls
set protocols mpls ipv6-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.1
set protocols bgp group ibgp family inet labeled-unicast per-prefix-label
set protocols bgp group ibgp family inet6 labeled-unicast explicit-null
set protocols bgp group ibgp export nhself
set protocols bgp group ibgp neighbor 192.168.0.4 description PE1
set protocols bgp group ibgp neighbor 192.168.0.5 description PE2
set protocols bgp group ebgp type external
set protocols bgp group ebgp local address 192.168.0.1
set protocols bgp group ebgp family inet labeled-unicast
set protocols bgp group ebgp family inet6 labeled-unicast
set protocols bgp group ebgp peer-as 64497
set protocols bgp group ebgp neighbor 172.16.0.2 description CE0
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 1000
set protocols ospf3 area 0.0.0.0 interface all
set protocols ospf3 area 0.0.0.0 interface fxp0.0 disable
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-0/0/1.0 metric 1000
```

```
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options policy-statement lb then load-balance per-packet
set policy-options policy-statement nhself then next-hop self
set routing-options protect core
set routing-options forwarding-table export lb
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64496
```

Router P0

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 description P0->PE0
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.6/24
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::3/32
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description P0->PE1
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.9/24
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::4/32
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local address 192.168.0.1
set protocols bgp group ibgp neighbor 192.168.0.4 description PE1
set protocols bgp group ibgp neighbor 192.168.0.5 description PE2
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64496
```

Router P1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/1 unit 0 description P1->PE0
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.2/24
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::5/32
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description P1->PE2
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.13/24
```

```
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::6/32
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local address 192.168.0.3
set protocols bgp group ibgp neighbor 192.168.0.1 description PE0
set protocols bgp group ibgp neighbor 192.168.0.5 description PE2
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64496
```

Router PE1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 description PE1->P0
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.10/24
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::7/32
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/1 unit 0 description PE1->CE1
set interfaces ge-0/0/1 unit 0 family inet address 172.16.1.1/30
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::12/32
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local address 192.168.0.4
set protocols bgp group ibgp family inet labeled-unicast per-prefix-label
set protocols bgp group ibgp family inet6 labeled-unicast explicit-null
set protocols bgp group ibgp export nhself
set protocols bgp group ibgp neighbor 192.168.0.1 description PE0
set protocols bgp group ibgp neighbor 192.168.0.5 description PE2
set protocols bgp group ebgp type external
set protocols bgp group ebgp local address 192.168.0.4
set protocols bgp group ebgp peer-as 64497
set protocols bgp group ebgp neighbor 172.16.1.2 description CE1
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 1000
set protocols ospf3 area 0.0.0.0 interface all
```

```
set protocols ospf3 area 0.0.0.0 interface fxp0.0 disable
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 1000
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options policy-statement PE1-v6-nh_CE1 from family inet6
set policy-options policy-statement PE1-v6-nh_CE1 then next-hop 2001:DB8::13
set policy-options policy-statement nhself then next-hop self
set routing-options router-id 192.168.0.4
set routing-options autonomous-system 64496
set routing-options static route 192.168.1.2 next-hop 172.16.1.2
```

Router PE2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 description PE2->P1
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.14/24
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::8/32
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/1 unit 0 description PE2->CE2
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.1/30
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::14/32
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols mpls ipv6-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local address 192.168.0.5
set protocols bgp group ibgp family inet labeled-unicast per-prefix-label
set protocols bgp group ibgp family inet6 labeled-unicast explicit-null
set protocols bgp group ibgp export nhself
set protocols bgp group ibgp neighbor 192.168.0.4 description PE1
set protocols bgp group ibgp neighbor 192.168.0.1 description PE0
set protocols bgp group ebgp type external
set protocols bgp group ebgp local address 192.168.0.5
set protocols bgp group ebgp peer-as 64497
set protocols bgp group ebgp family inet labeled-unicast
set protocols bgp group ebgp family inet6 labeled-unicast
set protocols bgp group ebgp neighbor 172.16.2.2 description CE2
```

```
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 1000
set protocols ospf3 area 0.0.0.0 interface all
set protocols ospf3 area 0.0.0.0 interface fxp0.0 disable
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 1000
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options policy-statement nhself then next-hop self
set routing-options router-id 192.168.0.5
set routing-options autonomous-system 64496
set routing-options static route 192.168.1.3 next-hop 172.16.2.2
```

Device CE0

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/2 unit 0 description CE0->PE0
set interfaces ge-0/0/2 unit 0 family inet address 172.16.0.2/30
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8::11/32
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
set protocols mpls interface all
set protocols bgp group ebgp type external
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp family inet labeled-unicast
set protocols bgp group ebgp family inet6 labeled-unicast
set protocols bgp group ebgp neighbor 172.16.0.1 description PE0
set protocols bgp group ebgp local-address 192.168.1.1
set routing-options autonomous-system 64497
set routing-options router-id 192.168.1.1
```

Device CE1

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/2 unit 0 description CE1->PE1
set interfaces ge-0/0/2 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8::13/32
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.1.2/32
```

```
set interfaces lo0 unit 0 family inet address 192.168.1.5/24
set protocols mpls interface all
set protocols bgp group ebgp type external
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp family inet labeled-unicast
set protocols bgp group ebgp family inet6 labeled-unicast
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 172.16.1.1 description PE1
set policy-options policy statement send-direct from protocol direct then accept
set routing-options autonomous-system 64497
set routing-options router-id 192.168.1.2
```

Device CE2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/2 unit 0 description CE2->PE2
set interfaces ge-0/0/2 unit 0 family inet address 172.16.2.2/30
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8::15/32
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.1.3/32
set protocols mpls interface all
set protocols bgp group ebgp type external
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp family inet labeled-unicast
set protocols bgp group ebgp family inet6 labeled-unicast
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 172.16.2.1 description PE2
set policy-options policy statement send-direct from protocol direct then accept
set routing-options autonomous-system 64497
set routing-options router-id 192.168.1.3
```

Configuring Device PE0

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 Junos OS CLI User Guide.

To configure Device PE0:

1. On routers with Modular Port Concentrators (MPCs), enable enhanced IP network services.

```
[edit chassis]
usr@PE0# set network-services enhanced-ip
```

2. Configure the device interfaces.

```
[edit interfaces]
user@PE0# set ge-0/0/0 unit 0 description PE0->P0
user@PE0# set ge-0/0/0 unit 0 family inet address 10.0.0.5/24
user@PE0# set ge-0/0/0 unit 0 family iso
user@PE0# set ge-0/0/0 unit 0 family inet6 address 2001:db8::1/32
user@PE0# set ge-0/0/0 unit 0 family mpls
user@PE0# set ge-0/0/1 unit 0 description PE0->P1
user@PE0# set ge-0/0/1 unit 0 family inet address 10.0.0.1/24
user@PE0# set ge-0/0/1 unit 0 family iso
user@PE0# set ge-0/0/1 unit 0 family inet6 address 2001:db8::2/32
user@PE0# set ge-0/0/1 unit 0 family mpls
user@PE0# set ge-0/0/2 unit 0 description PE0->CE0
user@PE0# set ge-0/0/2 unit 0 family inet address 172.16.0.1/30
user@PE0# set ge-0/0/2 unit 0 family inet6 address 2001:db8::10/32
user@PE0# set ge-0/0/2 unit 0 family mpls
```

3. Configure the loopback interface.

```
[edit interfaces]
user@PE0# set lo0 unit 0 family inet address 192.168.0.1/32
```

4. Configure MPLS and LDP on all interfaces excluding the management interface.

```
[edit protocols]
user@PE0# set mpls ipv6-tunneling
user@PE0# set mpls interface all
user@PE0# set mpls interface fxp0.0 disable
user@PE0# set ldp track-igp-metric
user@PE0# set ldp interface all
user@PE0# set ldp interface fxp0.0 disable
```

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

```
[edit protocols]
user@PE0# set ospf area 0.0.0.0 interface all
user@PE0# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE0# set ospf area 0.0.0.0 interface lo0.0 passive
user@PE0# set ospf area 0.0.0.0 interface ge-0/0/1.0 metric 1000
user@PE0# set ospf3 area 0.0.0.0 interface all
user@PE0# set ospf3 area 0.0.0.0 interface fxp0.0 disable
user@PE0# set ospf3 area 0.0.0.0 interface lo0.0 passive
user@PE0# set ospf3 area 0.0.0.0 interface ge-0/0/1.0 metric 1000
```

6. Configure IBGP connections with the other PE devices.

```
[edit protocols]
user@PE0# set bgp group ibgp type internal
user@PE0# set bgp group ibgp local-address 192.168.0.1
user@PE0# set bgp group ibgp family inet labeled-unicast per-prefix-label
user@PE0# set bgp group ibgp family inet6 labeled-unicast explicit-null
user@PE0# set bgp group ibgp export nhself
user@PE0# set bgp group ibgp neighbor 192.168.0.4 description PE1
user@PE0# set bgp group ibgp neighbor 192.168.0.5 description PE2
```

7. Configure EBGP connections with the customer devices.

```
[edit protocols]
user@PE0# set bgp group ebgp type external
user@PE0# set bgp group ebgp local address 192.168.0.1
user@PE0# set bgp group ebgp family inet labeled-unicast
user@PE0# set bgp group ebgp family inet6 labeled-unicast
user@PE0# set bgp group ebgp peer-as 64497
user@PE0# set bgp group ebgp neighbor 172.16.0.2 description CE0
```

8. Configure the load-balancing policy.

```
[edit policy-options]
user@PE0# set policy-statement lb then load-balance per-packet
```

9. Configure a next-hop self policy.

```
[edit policy-options]
user@PE0# set policy-statement nhself then next-hop self
```

10. Enable the BGP PIC edge feature.

```
[edit routing-options]
user@PE0# set protect core
```

11. Apply the load-balancing policy.

```
[edit routing-options]
user@PE0# set forwarding-table export lb
```

12. Assign the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@PE0# set router-id 192.168.0.2
user@PE0# set autonomous-system 64496
```

Results

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

```
[edit]
user@PE0# show chassis
network-services enhanced-ip;
```

```
[edit]
user@PE0# show interfaces
ge-0/0/0 {
    unit 0 {
        description PE0->P0;
```

```
        family inet {
            address 10.0.0.5/24;
        }
        family iso;
        family inet6 {
            address 2001:db8::1/32;
        }
        family mpls;
    }
}
ge-0/0/1 {
    unit 0 {
        description PE0->P1;
        family inet {
            address 10.0.0.1/24;
        }
        family iso;
        family inet6 {
            address 2001:db8::2/32;
        }
        family mpls;
    }
}
ge-0/0/2 {
    unit 0 {
        description PE0->CE0;
        family inet {
            address 172.16.0.1/30;
        }
        family inet6 {
            address 2001:db8::10/32;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
```

```
    }
}
```

```
[edit]
user@PE0# show protocols
mpls {
    ipv6-tunneling;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 192.168.0.1;
        family inet {
            labeled-unicast {
                per-prefix-label;
            }
        }
        family inet6 {
            labeled-unicast {
                explicit-null;
            }
        }
        export nhself;
        neighbor 192.168.0.4 {
            description PE1;
        }
        neighbor 192.168.0.5 {
            description PE2;
        }
    }
    group ebgp {
        type external;
        local-address 192.168.0.1;
        family inet {
            labeled-unicast;
        }
        family inet6 {
```

```
            labeled-unicast;
        }
        peer-as 64497;
        neighbor 172.16.0.2 {
            description CE0;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/1.0 {
            metric 1000;
        }
        interface fxp0.0 {
            disable;
        }
    }
}
ospf3 {
    area 0.0.0.0 {
        interface all;
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/1.0 {
            metric 1000;
        }
        interface fxp0.0 {
            disable;
        }
    }
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
```

```
    }
}
```

```
[edit]
user@PE1# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
policy-statement nhself {
    then {
        next-hop self;
    }
}
```

```
[edit]
user@PE0# show routing-options
protect core;
router-id 192.168.0.1;
autonomous system 64496
forwarding-table {
    export lb;
}
```

Verification

IN THIS SECTION

- Displaying Extensive Route Information | 744
- Displaying the Forwarding Table | 747

Confirm that the configuration is working properly.

Displaying Extensive Route Information

Purpose

Confirm that BGP PIC edge is working.

Action

From Device PE0, run the `show route extensive` command.

```
user@PE0> show route 192.168.1.5 extensive
inet.0: 236941 destinations, 630411 routes (236940 active, 0 holddown, 1 hidden)
20.1.1.1/32 (3 entries, 2 announced)
        State: <CalcForwarding>
TSI:
KRT in-kernel 192.168.1.5/24 -> {indirect(1048574), indirect(1048575)}
        @BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xafd09d0
                Next-hop reference count: 236886
                Source: 192.168.0.4
                Next hop type: Router, Next hop index: 623
                Next hop: 10.0.0.2 via ge-0/0/1.0, selected
                Session Id: 0x140
                Protocol next hop: 192.168.0.4
                Indirect next hop: 0xab3b980 1048574 INH Session ID: 0x144
                State: <Active Int Ext ProtectionPath ProtectionCand>
                Local AS:   64496 Peer AS:   64496
                Age: 1:11       Metric2: 2
                Validation State: unverified
                Task: BGP_100.192.168.0.5
                Announcement bits (1): 6-Resolve tree 2
                AS path: 64497 I
                Accepted MultipathUnequal
                Localpref: 100
                Router ID: 192.168.0.5
                Indirect next hops: 1
                                                Protocol next hop: 192.168.0.5 Metric:
2
                                                 Indirect next hop: 0xab3b980 1048574 INH
Session ID: 0x144
                                                Indirect path forwarding next hops:
```

```
1
                                                                Next hop type:
Router
                                                                Next hop: 10.0.0.2 via
ge-0/0/1.0
                                                                Session Id:
0x140
                                                192.168.0.5/32 Originating RIB:
inet.0
                                                    Metric: 2                       Node path
count: 1
                                                    Forwarding nexthops:
1
                                                                Nexthop: 10.0.0.2 via
ge-0/0/1.0
      BGP    Preference: 170/-101
                            Next hop type: Indirect, Next hop index: 0
                            Address: 0xafd0970
                            Next-hop reference count: 196735
                            Source: 192.168.0.4
                            Next hop type: Router, Next hop index: 624
                            Next hop: 10.0.0.6 via ge-0/0/0.0, selected
                            Session Id: 0x141
                              Protocol next hop: 192.168.0.4
                              Indirect next hop: 0xab3c240 1048575 INH Session ID:
0x145
                              State: <NotBest Int Ext ProtectionCand>
                              Inactive reason: Not Best in its group - IGP metric
                              Local AS:   100 Peer AS:   100
                              Age: 1:05       Metric2: 1001
                              Validation State: unverified
                              Task: BGP_100.192.168.0.4
                              AS path: 200 400 I
                              Accepted
                              Localpref: 100
                              Router ID: 192.168.0.4
                              Indirect next hops: 1
                                          Protocol next hop:  192.168.0.4 Metric:
1001
                                          Indirect next hop: 0xab3c240 1048575 INH Session ID:
0x145
                                          Indirect path forwarding next hops:
1
```

```
                                                        Next hop type:
Router
                                                        Next hop: 10.0.0.6 via
ge-0/0/0.0
                                                        Session Id:
0x141
                                        192.168.0.4/32 Originating RIB:
inet.0
                                            Metric: 1001                    Node path count:
1
                                            Forwarding nexthops:
1
                                                        Nexthop: 10.0.0.6 via
ge-0/0/0.0
#Multipath Preference: 255
                                    Next hop type: Indirect, Next hop index: 0
                                    Address: 0xd330f90
                                    Next-hop reference count: 304062
                                    Next hop type: Router, Next hop index: 623
                                    Next hop: 10.0.0.6 via ge-0/0/0.0, selected
                                    Session Id: 0x140
                                    Next hop type: Router, Next hop index: 624
                                    Next hop: 10.0.0.2 via ge-0/0/1.0
                                    Session Id: 0x141
                                    Protocol next hop: 192.168.0.4
                                    Indirect next hop: 0xab3b980 1048574 INH Session ID: 0x144
Weight 0x1
                                    Protocol next hop: 192.168.0.5
                                    Indirect next hop: 0xab3c240 1048575 INH Session ID: 0x145
Weight 0x4000
                                    State:   <ForwardinOnly Int Ext>
                                    Inactive reason: Forwarding use only
                                    Local AS:   64496
                                    Age: 1:05       Metric2: 2
                                    Validation State: unverified
                                    Task: RT
                                    Announcement bits (1): 0-KRT
                                    AS path: 64497 I
```

Meaning

Junos OS uses the next hops and the `weight` values to select a backup path when a link failure occurs. The next-hop weight has one of the following values:

- 0x1 indicates the primary path with active next hops.
- 0x4000 indicates the backup path with passive next hops.

Displaying the Forwarding Table

Purpose

Check the forwarding and kernel routing-table state by using the `show route forwarding-table` command.

Action

From Device PE0, run the `show route forwarding-table destination 192.168.1.5 extensive` command.

```
user@PE0> show route forwarding-table destination 192.168.1.5 extensive
  Routing table: default.inet [Index 0]
Internet:

Destination:  192.168.1.5/24
     Route type: user
     Route reference: 0                   Route interface-index: 0
     Multicast RPF nh index: 0
     Flags: sent to PFE
     Next-hop type: unilist               Index: 1048576  Reference: 7401
     Next-hop type: indirect              Index: 1048574  Reference:
2                                         Weight: 0x1
     Nexthop: 10.0.0.6
     Next-hop type: unicast               Index: 623      Reference: 8
     Next-hop interface: ge-0/0/0.0    Weight: 0x1
     Next-hop type: indirect              Index: 1048575  Reference:
2                                         Weight: 0x4000
     Nexthop: 10.0.0.2
     Next-hop type: unicast               Index: 624      Reference: 8
     Next-hop interface: ge-0/0/1.0    Weight: 0x4000
```

Meaning

Junos OS uses the next hops and the `weight` values to select a backup path when a link failure occurs. The next-hop weight has one of the following values:

- 0x1 indicates the primary path with active next hops.
- 0x4000 indicates the backup path with passive next hops.

SEE ALSO

Configuring BGP Prefix Independent Convergence for Inet | 723

Use Case for BGP Prefix Independent Convergence for Inet, Inet6, or Labeled Unicast | 722

BGP PIC Edge Using BGP Labeled Unicast Overview

IN THIS SECTION

- Benefits of BGP PIC Edge Using BGP Labeled Unicast | 748
- How does BGP Prefix Independent Convergence Work? | 749
- BGP PIC Edge Using BGP Labeled Unicast as the Transport Protocol | 749

This section talks about the benefits and overview of BGP PIC Edge using BGP labeled unicast as the transport protocol.

Benefits of BGP PIC Edge Using BGP Labeled Unicast

This feature provides the following benefits:

- Provides traffic protection in case of border (ABR and ASBR) node failures in multi-domain networks.
- Provides faster restoration of network connectivity and reduces traffic loss if the primary path becomes unavailable.

How does BGP Prefix Independent Convergence Work?

BGP Prefix Independent Convergence (PIC) improves BGP convergence on network node failures. BGP PIC creates and stores primary and backup paths for the indirect next hop on the Routing Engine and also provides the indirect next hop route information to the Packet Forwarding Engine. When a network node failure occurs, the Routing Engine signals the Packet Forwarding Engine that an indirect next hop has failed, and that the traffic is rerouted to a pre-calculated equal-cost or backup path without modifying BGP prefixes. Routing the traffic to the destination prefix continues by using the backup path to reduce traffic loss until the global convergence through BGP is resolved.

BGP convergence is applicable to both core and edge network node failures. In the case of BGP PIC Core, adjustments to the forwarding chains are made as a result of node or core link failures. In the case of BGP PIC Edge, adjustments to the forwarding chains are made as a result of edge node or edge link failures.

BGP PIC Edge Using BGP Labeled Unicast as the Transport Protocol

BGP PIC Edge using the BGP labeled unicast transport protocol helps to protect and reroute traffic when border nodes (ABR and ASBR) failures happen in multi-domain networks. Multi-domain networks are typically used in Metro Ethernet aggregation and mobile backhaul network designs.

On Juniper Networks MX Series, EX Series, and PTX Series devices, BGP PIC Edge supports Layer 3 services with BGP labeled unicast as the transport protocol. Additionally, on Juniper Networks MX Series, EX9204, EX9208, EX9214, EX9251, and EX9253 devices, BGP PIC Edge supports Layer 2 circuit, Layer 2 VPN, and VPLS (BGP VPLS, LDP VPLS and FEC 129 VPLS) services with BGP labeled unicast as transport protocol. These BGP services are multipath (learned from multiple PEs) and resolved through BGP labeled unicast routes, which could again be a multipath learnt from other ABRs. Transport protocols supported over BGP PIC Edge are RSVP, LDP, OSPF, and ISIS. Starting from Junos OS Release 20.2R1, MX Series, EX9204, EX9208, EX9214, EX9251, and EX9253 devices support BGP PIC Edge protection for Layer 2 circuit, Layer 2 VPN, and VPLS (BGP VPLS, LDP VPLS and FEC 129 VPLS) services with BGP labeled unicast as the transport protocol.

On Juniper Networks MX Series, EX Series and PTX Series devices, BGP PIC Edge protection with BGP labeled unicast as the transport is supported for the following services:

- IPv4 services over IPv4 BGP labeled unicast
- IPv6 BGP labeled unicast service over IPv4 BGP labeled unicast
- IPv4 Layer 3 VPN services over IPv4 BGP labeled unicast
- IPv6 Layer 3 VPN services over IPv4 BGP labeled unicast

On Juniper Networks MX Series and EX Series devices, BGP PIC Edge protection with BGP labeled unicast as the transport is supported for the following services:

- Layer 2 circuit services over IPv4 BGP labeled unicast
- Layer 2 VPN services over IPv4 BGP labeled unicast
- VPLS (BGP VPLS, LDP VPLS, and FEC 129 VPLS) services over IPv4 BGP labeled unicast

Configuring BGP PIC Edge Using BGP Labeled Unicast for Layer 2 Services

MX Series, EX9204, EX9208, EX9214, EX9251, and EX9253 devices support BGP PIC Edge protection for Layer 2 circuit, Layer 2 VPN, and VPLS (BGP VPLS, LDP VPLS and FEC 129 VPLS) services with BGP labeled unicast as the transport protocol. BGP PIC Edge using the BGP labeled unicast transport protocol helps to protect traffic failures over border nodes (ABR and ASBR) in multi-domain networks. Multi-domain networks are typically used in metro-aggregation and mobile backhaul networks designs.

A prerequisite for BGP PIC Edge protection is to program the Packet Forwarding Engine (PFE) with expanded next-hop hierarchy.

To enable expanded next-hop hierarchy for BGP labeled unicast family, you need to configure the following CLI configuration statement at the [`edit protocols`] hierarchy level:

```
[edit protocols]
user@host#set bgp group group-name family inet labeled-unicast nexthop-resolution preserve-
nexthop-hierarchy;
```

To enable BGP PIC for MPLS load balance nexthops, you need to configure the following CLI configuration statement at the [`edit routing-options`] hierarchy level:

```
[edit routing-options]
user@host#set rib routing-table-name protect core;
```

To enable fast convergence for Layer 2 services, you need to configure the following CLI configuration statements at the [`edit protocols`] hierarchy level:

For Layer 2 circuit and LDP VPLS:

```
[edit protocols]
user@host#set l2circuit resolution preserve-nexthop-heirarchy;
```

For Layer 2 VPN, BGP VPLS, and FEC129:

```
[edit protocols]
user@host#set l2vpn resolution preserve-nexthop-heirarchy;
```

Example: Protecting IPv4 Traffic over Layer 3 VPN Running BGP Labeled Unicast

IN THIS SECTION

- Requirements | 751
- Overview | 751
- Configuration | 753
- Verification | 819

This example shows how to configure BGP prefix-independent convergence (PIC) edge labeled unicast and protect IPv4 traffic over Layer 3 VPN. When an IPv4 traffic from a CE router is sent to a PE router, the IPv4 traffic is routed over a Layer 3 VPN, where BGP labeled unicast is configured as the transport protocol.

Requirements

This example uses the following hardware and software components:

- MX Series routers.
- Junos OS Release 19.4R1 or later running on all devices.

Overview

IN THIS SECTION

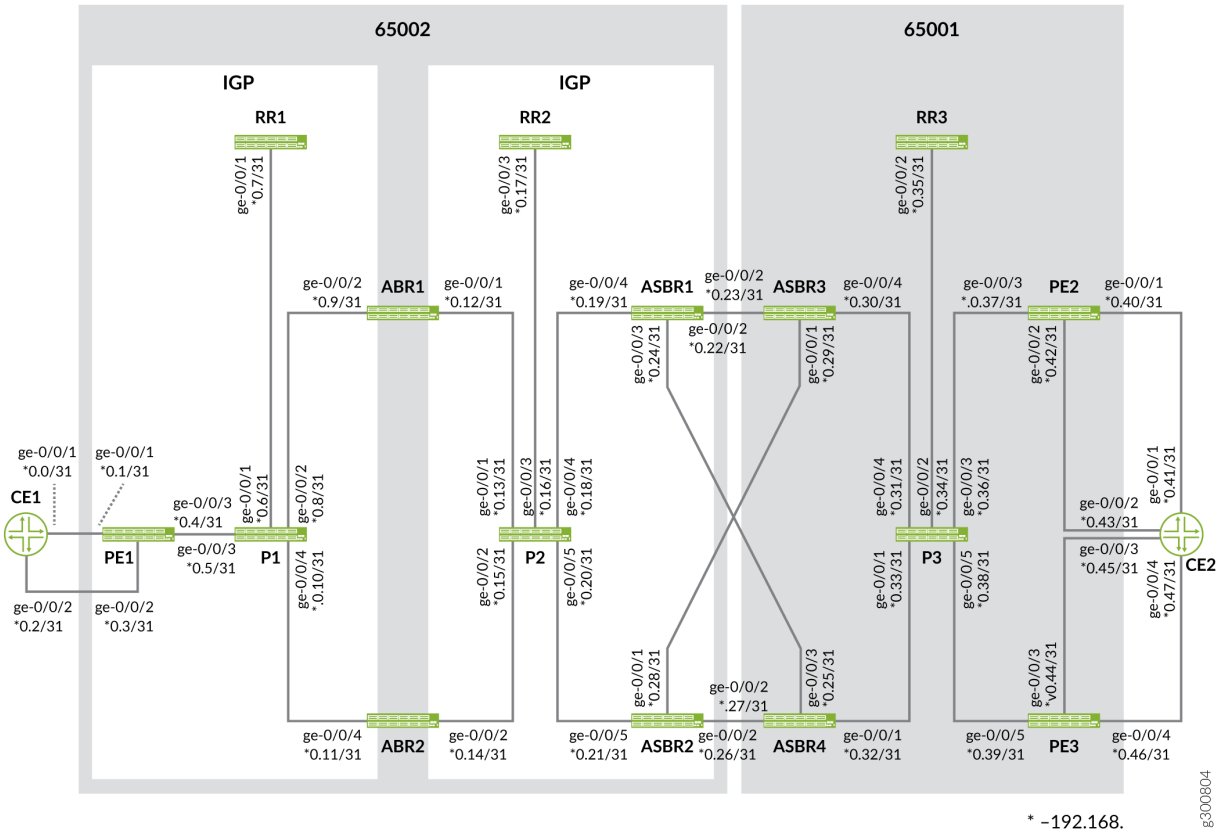
- Topology | 752

The following topology provides both ABR and ASBR protection by switching the traffic to backup paths whenever the primary path becomes unavailable.

Topology

Figure 55 on page 752 illustrates Layer 3 VPN running BGP labeled unicast as the inter-domain transport protocol.

Figure 55: Layer 3 VPN over BGP Labeled Unicast Using LDP Transport Protocol



The following table describes the components used in the topology:

Primary Components	Device Type	Position
CE1	MX Series	Connected to customer network.

(Continued)

Primary Components	Device Type	Position
PE1	MX Series	Configured with primary and backup routing paths to protect and reroute traffic from CE1 to CE2.
P1-P3	MX Series	Core routers to transport traffic.
ABR1-ABR2	MX Series	Area border routers
ABSR1-ABSR4	MX Series	Autonomous System Boundary Router
RR1-RR3	MX Series	Route Reflector
PE2-PE3	MX Series	PE routers connected to customer edge router (CE2).
CE2	MX Series	Connected to customer network.

PE2 and PE3 device addresses are learned from both ABR1 and ABR2 as labeled unicast routes. These routes are resolved over IGP/LDP protocols. PE1 learns CE2 routes from both PE2 and PE3 devices.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **754**
- Configuring CE1 | **777**
- Configuring PE1 | **780**
- Configuring P1 Device | **790**
- Configuring RR1 Device | **794**
- Configuring ABR1 Device | **799**
- Configuring ABR2 Device | **804**

Configuring P2 Device | 809
Configuring RR2 Device | 814

To configure BGP PIC Edge using BGP Label Unicast with LDP as the transport protocol, perform these tasks:

CLI Quick Configuration

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

Device CE1

```
set interfaces ge-0/0/1 description CE1-to-PE1-Link1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.0/31
set interfaces ge-0/0/2 description CE1-to-PE1-Link2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.2/31
set interfaces lo0 unit 0 family inet address 10.4.4.4/32
set policy-options policy-statement nhs term 1 from interface lo0.0
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set routing-options router-id 10.4.4.4
set routing-options autonomous-system 65004
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2 export nhs
set protocols bgp group toAs2 peer-as 65002
set protocols bgp group toAs2 neighbor 192.168.0.1
set protocols bgp group toAs2 neighbor 192.168.0.3
```

Device PE1

```
set interfaces ge-0/0/1 description PE1-to-CE1-Link1
set interfaces ge-0/0/1 vlan-tagging
```

```
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.1/31
set interfaces ge-0/0/2 description PE1-to-CE1-Link2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.3/31
set interfaces ge-0/0/3 description PE1-to-P1
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.4/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.2.2.5/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0005.00
set policy-options policy-statement add-noexport term 1 then community add noexport
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement mp-resolv term 1 from route-filter 10.1.1.0/24 orlonger
set policy-options policy-statement mp-resolv term 1 then accept
set policy-options policy-statement mp-resolv term 1 then multipath-resolve
set policy-options policy-statement mp-resolv term 2 from route-filter 10.2.2.0/24 orlonger
set policy-options policy-statement mp-resolv term 2 then accept
set policy-options policy-statement mp-resolv term 2 then multipath-resolve
set policy-options policy-statement mp-resolv term def then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then local-preference 65200
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement vrf-export-red term 1 then community add leak2red
set policy-options policy-statement vrf-export-red term 1 then accept
set policy-options policy-statement vrf-import-red term 1 from community leak2red
set policy-options policy-statement vrf-import-red term 1 then accept
set policy-options community leak2red members target:100:100
set policy-options community noexport members no-export
set policy-options community noexport members no-advertise
set routing-instances red routing-options multipath preserve-nexthop-hierarchy
set routing-instances red routing-options protect core
set routing-instances red protocols bgp group toCE1 peer-as 4
```

```
set routing-instances red protocols bgp group toCE1 neighbor 192.168.0.2
set routing-instances red instance-type vrf
set routing-instances red interface ge-0/0/2.0
set routing-instances red vrf-import vrf-import-red
set routing-instances red vrf-export vrf-export-red
set routing-options rib inet.3 protect core
set routing-options route-distinguisher-id 10.2.2.5
set routing-options forwarding-table export pplb
set routing-options resolution preserve-nexthop-hierarchy
set routing-options resolution rib inet.0 import mp-resolv
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.2.2.5
set routing-options autonomous-system 2
set routing-options protect core
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set routing-options rib-groups inet3to0 import-rib inet.3
set routing-options rib-groups inet3to0 import-rib inet.0
set routing-options rib-groups inet3to0 import-policy add-noexport
set protocols isis level 1 disable
set protocols isis interface ge-0/0/3.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/3.0
set protocols ldp interface ge-0/0/3.0
set protocols mpls label-switched-path toABR1-gold to 10.2.2.3
set protocols mpls label-switched-path toABR1-bronze to 10.2.2.3
set protocols mpls label-switched-path toABR2-gold to 10.2.2.4
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2RR type internal
set protocols bgp group toAs2RR local-address 10.2.2.5
set protocols bgp group toAs2RR family inet labeled-unicast rib-group inet3to0
set protocols bgp group toAs2RR family inet labeled-unicast add-path receive
set protocols bgp group toAs2RR family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs2RR family inet labeled-unicast nexthop-resolution preserve-nexthop-
hierarchy
set protocols bgp group toAs2RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs2RR export nhs
set protocols bgp group toAs2RR export export-inet3
set protocols bgp group toAs2RR neighbor 10.2.2.6
set protocols bgp group toAs4 peer-as 65004
set protocols bgp group toAs4 neighbor 192.168.0.0
```

```
set protocols bgp group toAs1PEs multihop no-nexthop-change
set protocols bgp group toAs1PEs local-address 10.2.2.5
set protocols bgp group toAs1PEs family inet unicast
set protocols bgp group toAs1PEs family inet-vpn unicast
set protocols bgp group toAs1PEs family inet6 unicast
set protocols bgp group toAs1PEs family inet6-vpn unicast
set protocols bgp group toAs1PEs export nhs
set protocols bgp group toAs1PEs peer-as 65001
set protocols bgp group toAs1PEs neighbor 10.1.1.1
set protocols bgp group toAs1PEs neighbor 10.1.1.2
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
set protocols bgp multipath
```

Device P1

```
set interfaces ge-0/0/1 description P1-to-RR1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.6/31
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 description P1-to-ABR1
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.8/31
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description P1-to-PE1
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.5/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description P1-to-ABR2
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.10/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
```

```
set interfaces lo0 unit 0 family inet address 10.2.2.8/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0008.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set routing-options router-id 10.2.2.8
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
```

Device RR1

```
set interfaces ge-0/0/1 description RR1-to-P1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.7/31
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.2.2.6/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0006.00
set policy-options policy-statement add-noexport term 1 then community add noexport
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community noexport members no-export
set policy-options community noexport members no-advertise
set routing-options forwarding-table export pplb
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.2.2.6
set routing-options autonomous-system 2
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set routing-options rib-groups inet3to0 import-rib inet.3
```

```
set routing-options rib-groups inet3to0 import-rib inet.0
set routing-options rib-groups inet3to0 import-rib inet6.3
set routing-options rib-groups inet3to0 import-policy add-noexport
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2Reg2BNs type internal
set protocols bgp group toAs2Reg2BNs family inet labeled-unicast rib-group inet3to0
set protocols bgp group toAs2Reg2BNs family inet labeled-unicast add-path receive
set protocols bgp group toAs2Reg2BNs family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs2Reg2BNs family inet labeled-unicast rib inet.3
set protocols bgp group toAs2Reg2BNs export export-inet3
set protocols bgp group toAs2Reg2BNs neighbor 10.2.2.3
set protocols bgp group toAs2Reg2BNs neighbor 10.2.2.4
set protocols bgp group toAs2Reg2BNs neighbor 10.2.2.5
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
set protocols bgp local-address 10.2.2.6
set protocols bgp cluster 10.2.2.6
```

Device ABR1

```
set interfaces ge-0/0/1 description ABR1-to-P2
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.12/31
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 description ABR1-to-P1
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.9/31
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 10.2.2.3/32
```

```
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0003.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.2.2.3
set routing-options autonomous-system 65002
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls label-switched-path toASBR2-gold to 10.2.2.2
set protocols mpls label-switched-path toASBR1-bronze to 10.2.2.1
set protocols mpls label-switched-path toASBR2-bronze to 10.2.2.2
set protocols mpls interface all
set protocols bgp group toAs2RR type internal
set protocols bgp group toAs2RR local-address 10.2.2.3
set protocols bgp group toAs2RR advertise-inactive
set protocols bgp group toAs2RR family inet labeled-unicast add-path receive
set protocols bgp group toAs2RR family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs2RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs2RR export nhs
set protocols bgp group toAs2RR cluster 10.2.2.3
set protocols bgp group toAs2RR neighbor 10.2.2.6
set protocols bgp group toAs2RR neighbor 10.2.2.7
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device ABR2

```
set interfaces ge-0/0/2 description ABR2-to-P2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.14/31
```

```
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/4 description ABR2-to-P1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.11/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.2.2.4/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0004.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.2.2.4
set routing-options autonomous-system 65002
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls label-switched-path toASBR1-bronze to 10.2.2.1
set protocols mpls interface all
set protocols bgp group toAs2RR type internal
set protocols bgp group toAs2RR local-address 10.2.2.4
set protocols bgp group toAs2RR advertise-inactive
set protocols bgp group toAs2RR family inet labeled-unicast add-path receive
set protocols bgp group toAs2RR family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs2RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs2RR export nhs
set protocols bgp group toAs2RR cluster 10.2.2.4
set protocols bgp group toAs2RR neighbor 10.2.2.6
set protocols bgp group toAs2RR neighbor 10.2.2.7
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device P2

```
set interfaces ge-0/0/1 description P2-to-ABR1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.13/31
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 description P2-to-ABR2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.15/31
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description P2-to-RR2
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.16/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description P2-to-ASBR1
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.18/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description P2-to-ASBR2
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 vlan-id 100
set interfaces ge-0/0/5 unit 0 family inet address 192.168.0.20/31
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.2.2.9/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0009.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set routing-options router-id 10.2.2.9
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
```

```
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
```

Device RR2

```
set interfaces ge-0/0/3 description RR2-to-P2
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.17/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.2.2.7/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0007.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.2.2.7
set routing-options autonomous-system 65002
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2Reg1BNs type internal
set protocols bgp group toAs2Reg1BNs family inet labeled-unicast add-path receive
set protocols bgp group toAs2Reg1BNs family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs2Reg1BNs family inet labeled-unicast rib inet.3
set protocols bgp group toAs2Reg1BNs neighbor 10.2.2.1
set protocols bgp group toAs2Reg1BNs neighbor 10.2.2.2
```

```
set protocols bgp group toAs2Reg1BNs neighbor 10.2.2.3
set protocols bgp group toAs2Reg1BNs neighbor 10.2.2.4
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
set protocols bgp local-address 10.2.2.7
set protocols bgp cluster 10.2.2.7
```

Device ASBR1

```
set interfaces ge-0/0/2 description ASBR1-to-ASBR3
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.22/31
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description ASBR1-to-ASBR4
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.24/31
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description ASBR1-to-P2
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.19/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.2.2.1/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0001.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.2.2.1
set routing-options autonomous-system 65002
set protocols isis level 1 disable
set protocols isis interface ge-0/0/4.0
set protocols isis export allow-lo0
```

```
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/4.0
set protocols ldp interface ge-0/0/4.0
set protocols mpls interface ge-0/0/4.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs1-T family inet labeled-unicast rib inet.3
set protocols bgp group toAs1-T peer-as 1
set protocols bgp group toAs1-T neighbor 192.168.0.23
set protocols bgp group toAs1-T neighbor 192.168.0.27
set protocols bgp group toAs2RR type internal
set protocols bgp group toAs2RR local-address 10.2.2.1
set protocols bgp group toAs2RR advertise-external
set protocols bgp group toAs2RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs2RR export nhs
set protocols bgp group toAs2RR neighbor 10.2.2.7
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device ASBR2

```
set interfaces ge-0/0/1 description ASBR2-to-ASBR3
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.28/31
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description ASBR2-to-ASBR4
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.26/31
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/5 description ASBR2-to-P2
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 vlan-id 100
set interfaces ge-0/0/5 unit 0 family inet address 192.168.0.21/31
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.2.2.2/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0002.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
```

```
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.2.2.2
set routing-options autonomous-system 65002
set protocols isis level 1 disable
set protocols isis interface ge-0/0/5.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/5.0
set protocols ldp interface ge-0/0/5.0
set protocols mpls interface ge-0/0/5.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs1-T family inet labeled-unicast rib inet.3
set protocols bgp group toAs1-T peer-as 1
set protocols bgp group toAs1-T neighbor 192.168.0.29
set protocols bgp group toAs1-T neighbor 192.168.0.25
set protocols bgp group toAs2RR type internal
set protocols bgp group toAs2RR local-address 10.2.2.2
set protocols bgp group toAs2RR advertise-external
set protocols bgp group toAs2RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs2RR export nhs
set protocols bgp group toAs2RR neighbor 10.2.2.7
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device ASBR3

```
set interfaces ge-0/0/1 description ASBR3-to-ASBR2
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.29/31
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description ASBR3-to-ASBR1
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.23/31
```

```
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/4 description ASBR3-to-P3
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.30/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.1.3/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0003.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.1.1.3
set routing-options autonomous-system 65001
set protocols isis level 1 disable
set protocols isis interface ge-0/0/4.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/4.0
set protocols ldp interface ge-0/0/4.0
set protocols mpls label-switched-path toPE2-gold to 10.1.1.1
set protocols mpls interface ge-0/0/4.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2-T family inet labeled-unicast rib inet.3
set protocols bgp group toAs2-T peer-as 65002
set protocols bgp group toAs2-T neighbor 192.168.0.22
set protocols bgp group toAs2-T neighbor 192.168.0.28
set protocols bgp group toAs1RR type internal
set protocols bgp group toAs1RR local-address 10.1.1.3
set protocols bgp group toAs1RR advertise-external
set protocols bgp group toAs1RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs1RR export nhs
set protocols bgp group toAs1RR neighbor 10.1.1.6
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device ASBR4

```
set interfaces ge-0/0/1 description ASBR4-to-P3
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.32/31
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 description ASBR4-to-ASBR2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.27/31
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description ASBR4-to-ASBR1
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.25/31
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.1.4/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0004.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set routing-options forwarding-table export pplb
set routing-options router-id 10.1.1.4
set routing-options autonomous-system 65001
set protocols isis level 1 disable
set protocols isis interface ge-0/0/1.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/1.0
set protocols ldp interface ge-0/0/1.0
set protocols mpls label-switched-path toPE2-bronze to 10.1.1.1
set protocols mpls label-switched-path toPE3-bronze to 10.1.1.2
set protocols mpls interface ge-0/0/1.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs2-T family inet labeled-unicast rib inet.3
set protocols bgp group toAs2-T peer-as 2
```

```
set protocols bgp group toAs2-T neighbor 192.168.0.26
set protocols bgp group toAs2-T neighbor 192.168.0.24
set protocols bgp group toAs1RR type internal
set protocols bgp group toAs1RR local-address 10.1.1.4
set protocols bgp group toAs1RR advertise-external
set protocols bgp group toAs1RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs1RR export nhs
set protocols bgp group toAs1RR neighbor 10.1.1.6
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device RR3

```
set interfaces ge-0/0/2 description RR3-to-P3
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.35/31
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.1.1.6/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0006.00
set policy-options policy-statement add-noexport term 1 then community add noexport
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community noexport members no-export
set policy-options community noexport members no-advertise
set routing-options forwarding-table export pplb
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.1.1.6
set routing-options autonomous-system 1
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set routing-options rib-groups inet3to0 import-rib inet.3
set routing-options rib-groups inet3to0 import-rib inet.0
```

```
set routing-options rib-groups inet3to0 import-policy add-noexport
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
set protocols bgp path-selection external-router-id
set protocols bgp group toAs1BNs type internal
set protocols bgp group toAs1BNs family inet labeled-unicast rib-group inet3to0
set protocols bgp group toAs1BNs family inet labeled-unicast add-path receive
set protocols bgp group toAs1BNs family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs1BNs family inet labeled-unicast rib inet.3
set protocols bgp group toAs1BNs export export-inet3
set protocols bgp group toAs1BNs neighbor 10.1.1.3
set protocols bgp group toAs1BNs neighbor 10.1.1.4
set protocols bgp group toAs1BNs neighbor 10.1.1.2
set protocols bgp group toAs1BNs neighbor 10.1.1.1
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
set protocols bgp local-address 10.1.1.6
set protocols bgp cluster 10.1.1.6
```

Device P3

```
set interfaces ge-0/0/1 description P3-to-ASBR4
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.33/31
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 description P3-to-RR3
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.34/31
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 description P3-to-PE2
set interfaces ge-0/0/3 vlan-tagging
```

```
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.36/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 description P3-to-ASBR3
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.31/31
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/0/5 description P3-to-PE3
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 vlan-id 100
set interfaces ge-0/0/5 unit 0 family inet address 192.168.0.38/31
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.1.5/32
set interfaces lo0 unit 0 family iso address 49.0000.0000.aaaa.0005.00
set policy-options policy-statement allow-lo0 term 1 from interface lo0.0
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set routing-options router-id 10.1.1.5
set protocols isis level 1 disable
set protocols isis interface all
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface all
set protocols ldp interface all
set protocols mpls interface all
```

Device PE2

```
set interfaces ge-0/0/1 description PE2-to-CE2-Link1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.40/31
set interfaces ge-0/0/2 description PE2-to-CE2-Link2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.42/31
set interfaces ge-0/0/3 description PE2-to-P3
set interfaces ge-0/0/3 vlan-tagging
```

```
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.37/31
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.1.1.1/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0001.00
set policy-options policy-statement add-noexport term 1 then community add noexport
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then metric add 0
set policy-options policy-statement nhs term 1 then local-preference 200
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement vrf-export-red term 1 then community add leak2red
set policy-options policy-statement vrf-export-red term 1 then accept
set policy-options policy-statement vrf-import-red term 1 from community leak2red
set policy-options policy-statement vrf-import-red term 1 then accept
set policy-options community leak2red members target:100:100
set policy-options community noexport members no-export
set policy-options community noexport members no-advertise
set routing-instances red protocols bgp group toCE2 peer-as 65003
set routing-instances red protocols bgp group toCE2 neighbor 192.168.0.43
set routing-instances red instance-type vrf
set routing-instances red interface ge-0/0/2.0
set routing-instances red vrf-import vrf-import-red
set routing-instances red vrf-export vrf-export-red
set routing-options route-distinguisher-id 10.1.1.1
set routing-options forwarding-table export pplb
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.1.1.1
set routing-options autonomous-system 1
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set routing-options rib-groups inet3to0 import-rib inet.3
set routing-options rib-groups inet3to0 import-rib inet.0
set routing-options rib-groups inet3to0 import-policy add-noexport
```

```
set protocols isis level 1 disable
set protocols isis interface ge-0/0/3.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/3.0
set protocols ldp interface ge-0/0/3.0
set protocols mpls interface ge-0/0/3.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs3-1 peer-as 65003
set protocols bgp group toAs3-1 neighbor 192.168.0.41
set protocols bgp group toAs1RR type internal
set protocols bgp group toAs1RR local-address 10.1.1.1
set protocols bgp group toAs1RR advertise-external
set protocols bgp group toAs1RR family inet labeled-unicast rib-group inet3to0
set protocols bgp group toAs1RR family inet labeled-unicast add-path receive
set protocols bgp group toAs1RR family inet labeled-unicast add-path send path-count 4
set protocols bgp group toAs1RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs1RR family inet6-vpn unicast
set protocols bgp group toAs1RR export nhs
set protocols bgp group toAs1RR export export-inet3
set protocols bgp group toAs1RR neighbor 10.1.1.6
set protocols bgp group toAs2PEs multihop no-nexthop-change
set protocols bgp group toAs2PEs local-address 10.1.1.1
set protocols bgp group toAs2PEs family inet unicast
set protocols bgp group toAs2PEs family inet-vpn unicast
set protocols bgp group toAs2PEs family inet6 unicast
set protocols bgp group toAs2PEs family inet6-vpn unicast
set protocols bgp group toAs2PEs export nhs
set protocols bgp group toAs2PEs peer-as 65002
set protocols bgp group toAs2PEs neighbor 10.2.2.5
set protocols bgp group toAs2PEs vpn-apply-export
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device PE3

```
set interfaces ge-0/0/3 description PE3-to-CE2-Link1
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.44/31
```

```
set interfaces ge-0/0/4 description PE3-to-CE2-Link2
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.46/31
set interfaces ge-0/0/5 description PE3-to-P3
set interfaces ge-0/0/5 vlan-tagging
set interfaces ge-0/0/5 unit 0 vlan-id 100
set interfaces ge-0/0/5 unit 0 family inet address 192.168.0.39/31
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.1.1.2/32
set interfaces lo0 unit 1 family iso address 49.0000.0000.aaaa.0002.00
set policy-options policy-statement add-noexport term 1 then community add noexport
set policy-options policy-statement allow-lo0 term 1 from interface lo0.1
set policy-options policy-statement allow-lo0 term 1 then accept
set policy-options policy-statement allow-lo0 term 2 then reject
set policy-options policy-statement export-inet3 term 1 from rib inet.3
set policy-options policy-statement export-inet3 term 1 then accept
set policy-options policy-statement export-inet3 term 2 then reject
set policy-options policy-statement nhs term 1 from protocol bgp
set policy-options policy-statement nhs term 1 then metric add 0
set policy-options policy-statement nhs term 1 then local-preference 200
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options policy-statement vrf-export-red term 1 then community add leak2red
set policy-options policy-statement vrf-export-red term 1 then accept
set policy-options policy-statement vrf-import-red term 1 from community leak2red
set policy-options policy-statement vrf-import-red term 1 then accept
set policy-options community leak2red members target:100:100
set policy-options community noexport members no-export
set policy-options community noexport members no-advertise
set routing-instances red protocols bgp group toCE2 peer-as 65003
set routing-instances red protocols bgp group toCE2 neighbor 192.168.0.47
set routing-instances red instance-type vrf
set routing-instances red interface ge-0/0/4.0
set routing-instances red vrf-import vrf-import-red
set routing-instances red vrf-export vrf-export-red
set routing-options route-distinguisher-id 10.1.1.2
set routing-options forwarding-table export pplb
set routing-options interface-routes rib-group inet inet0to3
set routing-options router-id 10.1.1.2
set routing-options autonomous-system 1
```

```
set routing-options rib-groups inet0to3 import-rib inet.0
set routing-options rib-groups inet0to3 import-rib inet.3
set routing-options rib-groups inet0to3 import-policy allow-lo0
set routing-options rib-groups inet3to0 import-rib inet.3
set routing-options rib-groups inet3to0 import-rib inet.0
set routing-options rib-groups inet3to0 import-policy add-noexport
set protocols isis level 1 disable
set protocols isis interface ge-0/0/5.0
set protocols isis export allow-lo0
set protocols isis topologies ipv6-unicast
set protocols rsvp interface ge-0/0/5.0
set protocols ldp interface ge-0/0/5.0
set protocols mpls interface ge-0/0/5.0
set protocols bgp path-selection external-router-id
set protocols bgp group toAs3 peer-as 65003
set protocols bgp group toAs3 neighbor 192.168.0.45
set protocols bgp group toAs1RR type internal
set protocols bgp group toAs1RR local-address 10.1.1.2
set protocols bgp group toAs1RR advertise-external
set protocols bgp group toAs1RR family inet labeled-unicast rib-group inet3to0
set protocols bgp group toAs1RR family inet labeled-unicast rib inet.3
set protocols bgp group toAs1RR export nhs
set protocols bgp group toAs1RR export export-inet3
set protocols bgp group toAs1RR neighbor 10.1.1.6
set protocols bgp group toAs2PEs multihop no-nexthop-change
set protocols bgp group toAs2PEs local-address 10.1.1.2
set protocols bgp group toAs2PEs family inet unicast
set protocols bgp group toAs2PEs family inet-vpn unicast
set protocols bgp group toAs2PEs family inet6 unicast
set protocols bgp group toAs2PEs family inet6-vpn unicast
set protocols bgp group toAs2PEs export nhs
set protocols bgp group toAs2PEs peer-as 65002
set protocols bgp group toAs2PEs neighbor 10.2.2.5
set protocols bgp group toAs2PEs vpn-apply-export
set protocols bgp traceoptions file bgp.log
set protocols bgp traceoptions file size 100m
set protocols bgp traceoptions flag state detail
set protocols bgp traceoptions flag policy
```

Device CE2

```
set interfaces ge-0/0/1 description CE2-to-PE2-Link1
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 100
set interfaces ge-0/0/1 unit 0 family inet address 192.168.0.41/31
set interfaces ge-0/0/2 description CE2-to-PE2-Link2
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 100
set interfaces ge-0/0/2 unit 0 family inet address 192.168.0.43/31
set interfaces ge-0/0/3 description CE2-to-PE3-Link1
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 unit 0 vlan-id 100
set interfaces ge-0/0/3 unit 0 family inet address 192.168.0.45/31
set interfaces ge-0/0/4 description CE2-to-PE3-Link2
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 unit 0 vlan-id 100
set interfaces ge-0/0/4 unit 0 family inet address 192.168.0.47/31
set interfaces lo0 unit 0 family inet address 10.3.3.3/32
set policy-options policy-statement nhs term 1 from interface lo0.0
set policy-options policy-statement nhs term 1 then metric 50
set policy-options policy-statement nhs term 1 then next-hop self
set policy-options policy-statement nhs term 1 then accept
set policy-options policy-statement nhsMED100 term 1 from interface lo0.0
set policy-options policy-statement nhsMED100 term 1 then metric 100
set policy-options policy-statement nhsMED100 term 1 then next-hop self
set policy-options policy-statement nhsMED100 term 1 then accept
set policy-options community map2bronze members 100:200
set policy-options community map2gold members 100:100
set policy-options community map2gold_bronze_plain members 300:400
set routing-options router-id 10.3.3.3
set routing-options autonomous-system 65003
set protocols bgp path-selection external-router-id
set protocols bgp group toAs1Internet export nhs
set protocols bgp group toAs1Internet peer-as 65001
set protocols bgp group toAs1Internet neighbor 192.168.0.40
set protocols bgp group toAs1Internet neighbor 192.168.0.44 export nhsMED100
set protocols bgp group toAs1L3VPN export nhs
set protocols bgp group toAs1L3VPN peer-as 65001
set protocols bgp group toAs1L3VPN neighbor 192.168.0.46
set protocols bgp group toAs1L3VPN neighbor 192.168.0.42 export nhsMED100
```

Configuring CE1

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 CE1:

1. Configure the interfaces to enable IP and MPLS transport.

```
[edit interfaces]
user@CE1#set ge-0/0/1 description CE1-to-PE1-Link1
user@CE1#set ge-0/0/1 vlan-tagging
user@CE1#set ge-0/0/1 unit 0 vlan-id 100
user@CE1#set ge-0/0/1 unit 0 family inet address 192.168.0.0/31
user@CE1#set ge-0/0/2 description CE1-to-PE1-Link2
user@CE1#set ge-0/0/2 vlan-tagging
user@CE1#set ge-0/0/2 unit 0 vlan-id 100
user@CE1#set ge-0/0/2 unit 0 family inet address 192.168.0.2/31
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@CE1#set lo0 unit 0 family inet address 10.4.4.4/32
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@CE1#set policy-statement nhs term 1 from interface lo0.0
user@CE1#set policy-statement nhs term 1 then next-hop self
user@CE1#set policy-statement nhs term 1 then accept
```

4. Configure routing options.

```
[edit routing-options]
user@CE1#set router-id 10.4.4.4
user@CE1#set autonomous-system 65004
```

5. Configure BGP labeled unicast to ABRs to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols bgp]
user@CE1#set path-selection external-router-id
user@CE1#set group toAs2 export nhs
user@CE1#set group toAs2 peer-as 65002
user@CE1#set group toAs2 neighbor 192.168.0.1
user@CE1#set group toAs2 neighbor 192.168.0.3
```

Results

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

```
interfaces {
    ge-0/0/1 {
        description CE1-to-PE1-Link1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.0/31;
            }
        }
    }
    ge-0/0/2 {
        description CE1-to-PE1-Link2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
```

```
                address 192.168.0.2/31;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.4.4.4/32;
            }
        }
    }
}
policy-options {
    policy-statement nhs {
        term 1 {
            from interface lo0.0;
            then {
                next-hop self;
                accept;
            }
        }
    }
}
routing-options {
    router-id 10.4.4.4;
    autonomous-system 65004;
}
protocols {
    bgp {
        path-selection external-router-id;
        group toAs2 {
            export nhs;
            peer-as 65002;
            neighbor 192.168.0.1;
            neighbor 192.168.0.3;
        }
    }
}
```

Configuring 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 to enable IP and MPLS transport.

```
[edit interfaces]
user@PE1#set ge-0/0/1 description PE1-to-CE1-Link1
user@PE1#set ge-0/0/1 vlan-tagging
user@PE1#set ge-0/0/1 unit 0 vlan-id 100
user@PE1#set ge-0/0/1 unit 0 family inet address 192.168.0.1/31
user@PE1#set ge-0/0/2 description PE1-to-CE1-Link2
user@PE1#set ge-0/0/2 vlan-tagging
user@PE1#set ge-0/0/2 unit 0 vlan-id 100
user@PE1#set ge-0/0/2 unit 0 family inet address 192.168.0.3/31
user@PE1#set ge-0/0/3 description PE1-to-P1
user@PE1#set ge-0/0/3 vlan-tagging
user@PE1#set ge-0/0/3 unit 0 vlan-id 100
user@PE1#set ge-0/0/3 unit 0 family inet address 192.168.0.4/31
user@PE1#set ge-0/0/3 unit 0 family iso
user@PE1#set ge-0/0/3 unit 0 family mpls
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@PE1#set lo0 unit 1 family inet address 10.2.2.5/32
user@PE1#set lo0 unit 1 family iso address 49.0000.0000.aaaa.0005.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@PE1#set policy-statement add-noexport term 1 then community add noexport
user@PE1#set policy-statement allow-lo0 term 1 from interface lo0.1
user@PE1#set policy-statement allow-lo0 term 1 then accept
```

```
user@PE1#set policy-statement allow-lo0 term 2 then reject
user@PE1#set policy-statement export-inet3 term 1 from rib inet.3
user@PE1#set policy-statement export-inet3 term 1 then accept
user@PE1#set policy-statement export-inet3 term 2 then reject
user@PE1#set policy-statement mp-resolv term 1 from route-filter 10.1.1.0/24 orlonger
user@PE1#set policy-statement mp-resolv term 1 then accept
user@PE1#set policy-statement mp-resolv term 1 then multipath-resolve
user@PE1#set policy-statement mp-resolv term 2 from route-filter 10.2.2.0/24 orlonger
user@PE1#set policy-statement mp-resolv term 2 then accept
user@PE1#set policy-statement mp-resolv term 2 then multipath-resolve
user@PE1#set policy-statement mp-resolv term def then reject
user@PE1#set policy-statement nhs term 1 from protocol bgp
user@PE1#set policy-statement nhs term 1 then local-preference 200
user@PE1#set policy-statement nhs term 1 then next-hop self
user@PE1#set policy-statement nhs term 1 then accept
user@PE1#set policy-statement pplb then load-balance per-packet
user@PE1#set policy-statement vrf-export-red term 1 then community add leak2red
user@PE1#set policy-statement vrf-export-red term 1 then accept
user@PE1#set policy-statement vrf-import-red term 1 from community leak2red
user@PE1#set policy-statement vrf-import-red term 1 then accept
user@PE1#set community leak2red members target:100:100
user@PE1#set community noexport members no-export
user@PE1#set community noexport members no-advertise
```

4. Configure Layer 3 VPN routing instance to provide customer services.

```
[edit routing-instances]
user@PE1#set red routing-options multipath preserve-nexthop-hierarchy
user@PE1#set red routing-options protect core
user@PE1#set red protocols bgp group toCE1 peer-as 65004
user@PE1#set red protocols bgp group toCE1 neighbor 192.168.0.2
user@PE1#set red instance-type vrf
user@PE1#set red interface ge-0/0/2.0
user@PE1#set red vrf-import vrf-import-red
user@PE1#set red vrf-export vrf-export-red
```

5. Configure resolver RIB import policies and resolution RIBs to enable expanded hierarchical nexthop structure for selected Layer 3 VPN prefixes specified in the policy.

```
[edit routing-options]
user@PE1#set rib inet.3 protect core
```

```
user@PE1#set route-distinguisher-id 10.2.2.5
user@PE1#set forwarding-table export pplb
user@PE1#set resolution preserve-nexthop-hierarchy
user@PE1#set resolution rib inet.0 import mp-resolv
user@PE1#set interface-routes rib-group inet inet0to3
user@PE1#set router-id 10.2.2.5
user@PE1#set autonomous-system 65002
user@PE1#set protect core
user@PE1#set rib-groups inet0to3 import-rib inet.0
user@PE1#set rib-groups inet0to3 import-rib inet.3
user@PE1#set rib-groups inet0to3 import-policy allow-lo0
user@PE1#set rib-groups inet3to0 import-rib inet.3
user@PE1#set rib-groups inet3to0 import-rib inet.0
user@PE1#set rib-groups inet3to0 import-policy add-noexport
```

6. Configure OSPF protocol.

```
[edit protocols ospf]
user@PE1#set protocols ospf area 0.0.0.0 interface all link-protection;
user@PE1#set protocols ospf area 0.0.0.0 interface fxp0.0 disable;
user@PE1#set protocols ospf area 0.0.0.0 interface lo0.0 passive;
```

7. Configure routing protocols to establish IP and MPLS connectivity across the domain.

```
[edit protocols]
user@PE1#set isis level 1 disable
user@PE1#set isis interface ge-0/0/3.0
user@PE1#set isis export allow-lo0
user@PE1#set isis topologies ipv6-unicast
user@PE1#set rsvp interface ge-0/0/3.0
user@PE1#set ldp interface ge-0/0/3.0
user@PE1#set mpls label-switched-path toABR1-gold to 10.2.2.3
user@PE1#set mpls label-switched-path toABR1-bronze to 10.2.2.3
user@PE1#set mpls label-switched-path toABR2-gold to 10.2.2.4
```

8. Configure BGP labeled unicast to ABRs to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols bgp]
user@PE1#set path-selection external-router-id
```

```
user@PE1#set group toAs2RR type internal
user@PE1#set group toAs2RR local-address 10.2.2.5
user@PE1#set group toAs2RR family inet labeled-unicast rib-group inet3to0
user@PE1#set group toAs2RR family inet labeled-unicast add-path receive
user@PE1#set group toAs2RR family inet labeled-unicast add-path send path-count 4
user@PE1#set group toAs2RR family inet labeled-unicast nexthop-resolution preserve-nexthop-
hierarchy
user@PE1#set group toAs2RR family inet labeled-unicast rib inet.3
user@PE1#set group toAs2RR export nhs
user@PE1#set group toAs2RR export export-inet3
user@PE1#set group toAs2RR neighbor 10.2.2.6
user@PE1#set group toAs4 peer-as 65004
user@PE1#set group toAs4 neighbor 192.168.0.0
user@PE1#set group toAs1PEs multihop no-nexthop-change
user@PE1#set group toAs1PEs local-address 10.2.2.5
user@PE1#set group toAs1PEs family inet unicast
user@PE1#set group toAs1PEs family inet-vpn unicast
user@PE1#set group toAs1PEs family inet6 unicast
user@PE1#set group toAs1PEs family inet6-vpn unicast
user@PE1#set group toAs1PEs export nhs
user@PE1#set group toAs1PEs peer-as 65001
user@PE1#set group toAs1PEs neighbor 10.1.1.1
user@PE1#set group toAs1PEs neighbor 10 .1.1.2
user@PE1#set traceoptions file bgp.log
user@PE1#set traceoptions file size 100m
user@PE1#set traceoptions flag state detail
user@PE1#set traceoptions flag policy
user@PE1#set multipath
```

Results

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

```
interfaces {
    ge-0/0/1 {
        description PE1-to-CE1-Link1;
        vlan-tagging;
        unit 0 {
```

```
            vlan-id 100;
            family inet {
                address 192.168.0.1/31;
            }
        }
    }
    ge-0/0/2 {
        description PE1-to-CE1-Link2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.3/31;
            }
        }
    }
    ge-0/0/3 {
        description PE1-to-P1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.4/31;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 10.2.2.5/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0005.00;
            }
        }
    }
}
policy-options {
    policy-statement add-noexport {
        term 1 {
            then {
```

```
                community add noexport;
            }
        }
    }
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.1;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement export-inet3 {
        term 1 {
            from rib inet.3;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement mp-resolv {
        term 1 {
            from {
                route-filter 10.1.1.0/24 orlonger;
            }
            then {
                accept;
                multipath-resolve;
            }
        }
        term 2 {
            from {
                route-filter 10.2.2.0/24 orlonger;
            }
            then {
                accept;
                multipath-resolve;
            }
        }
        term def {
            then reject;
```

```
        }
    }
    policy-statement nhs {
        term 1 {
            from protocol bgp;
            then {
                local-preference 200;
                next-hop self;
                accept;
            }
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    policy-statement vrf-export-red {
        term 1 {
            then {
                community add leak2red;
                accept;
            }
        }
    }
    policy-statement vrf-import-red {
        term 1 {
            from community leak2red;
            then accept;
        }
    }
    community leak2red members target:100:100;
    community noexport members [ no-export no-advertise ];
}
routing-instances {
    red {
        routing-options {
            multipath preserve-nexthop-hierarchy;
            protect core;
        }
        protocols {
            bgp {
                group toCE1 {
```

```
                    peer-as 65004;
                    neighbor 192.168.0.2;
                }
            }
        }
        instance-type vrf;
        interface ge-0/0/2.0;
        vrf-import vrf-import-red;
        vrf-export vrf-export-red;
    }
}
routing-options {
    rib inet.3 {
        protect core;
    }
    route-distinguisher-id 10.2.2.5;
    forwarding-table {
        export pplb;
    }
    resolution {
        preserve-nexthop-hierarchy;
        rib inet.0 {
            import mp-resolv;
        }
    }
    interface-routes {
        rib-group inet inet0to3;
    }
    router-id 10.2.2.5;
    autonomous-system 65002;
    protect core;
    rib-groups {
        inet0to3 {
            import-rib [ inet.0 inet.3 ];
            import-policy allow-lo0;
        }
        inet3to0 {
            import-rib [ inet.3 inet.0 ];
            import-policy add-noexport;
        }
    }
}
protocols {
```

```
    isis {
        level 1 disable;
        interface ge-0/0/3.0;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface ge-0/0/3.0;
    }
    bgp {
        path-selection external-router-id;
        group toAs2RR {
            type internal;
            local-address 10.2.2.5;
            family inet {
                labeled-unicast {
                    rib-group inet3to0;
                    add-path {
                        receive;
                        send {
                            path-count 4;
                        }
                    }
                    nexthop-resolution {
                        preserve-nexthop-hierarchy;
                    }
                    rib {
                        inet.3;
                    }
                }
            }
            export [ nhs export-inet3 ];
            neighbor 10.2.2.6;
        }
        group toAs4 {
            peer-as 65004;
            neighbor 192.168.0.0;
        }
        group toAs1PEs {
            multihop {
                no-nexthop-change;
            }
            local-address 10.2.2.5;
```

```
            family inet {
                unicast;
            }
            family inet-vpn {
                unicast;
            }
            family inet6 {
                unicast;
            }
            family inet6-vpn {
                unicast;
            }
            export nhs;
            peer-as 65001;
            neighbor 10.1.1.1;
            neighbor 10.1.1.2;
        }
        traceoptions {
            file bgp.log size 100m;
            flag state detail;
            flag policy;
        }
        multipath;
    }
    ldp {
        interface ge-0/0/3.0;
    }
    mpls {
        label-switched-path toABR1-gold {
            to 10.2.2.3;
        }
        label-switched-path toABR1-bronze {
            to 10.2.2.3;
        }
        label-switched-path toABR2-gold {
            to 10.2.2.4;
        }
    }
}
```

Configuring P1 Device

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 P1:

1. Configure the interfaces.

```
[edit interfaces]
user@P1#set ge-0/0/1 description P1-to-RR1
user@P1#set ge-0/0/1 vlan-tagging
user@P1#set ge-0/0/1 unit 0 vlan-id 100
user@P1#set ge-0/0/1 unit 0 family inet address 192.168.0.6/31
user@P1#set ge-0/0/1 unit 0 family iso
user@P1#set ge-0/0/1 unit 0 family mpls
user@P1#set ge-0/0/2 description P1-to-ABR1
user@P1#set ge-0/0/2 vlan-tagging
user@P1#set ge-0/0/2 unit 0 vlan-id 100
user@P1#set ge-0/0/2 unit 0 family inet address 192.168.0.8/31
user@P1#set ge-0/0/2 unit 0 family iso
user@P1#set ge-0/0/2 unit 0 family mpls
user@P1#set ge-0/0/3 description P1-to-PE1
user@P1#set ge-0/0/3 vlan-tagging
user@P1#set ge-0/0/3 unit 0 vlan-id 100
user@P1#set ge-0/0/3 unit 0 family inet address 192.168.0.5/31
user@P1#set ge-0/0/3 unit 0 family iso
user@P1#set ge-0/0/3 unit 0 family mpls
user@P1#set ge-0/0/4 description P1-to-ABR2
user@P1#set ge-0/0/4 vlan-tagging
user@P1#set ge-0/0/4 unit 0 vlan-id 100
user@P1#set ge-0/0/4 unit 0 family inet address 192.168.0.10/31
user@P1#set ge-0/0/4 unit 0 family iso
user@P1#set ge-0/0/4 unit 0 family mpls
```

2. Configure the loopback interface.

```
[edit interfaces]
user@P1#set lo0 unit 0 family inet address 10.2.2.8/32
user@P1#set lo0 unit 0 family iso address 49.0000.0000.aaaa.0008.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@P1#set policy-statement allow-lo0 term 1 from interface lo0.0
user@P1#set policy-statement allow-lo0 term 1 then accept
user@P1#set policy-statement allow-lo0 term 2 then reject
```

4. Configure routing options.

```
[edit routing-options]
user@P1#set router-id 10.2.2.8
```

5. Configure ISIS, RSVP, LDP, and MPLS protocols on the interface.

```
[edit protocols]
user@P1#set isis level 1 disable
user@P1#set isis interface all
user@P1#set isis export allow-lo0
user@P1#set isis topologies ipv6-unicast
user@P1#set rsvp interface all
user@P1#set ldp interface all
user@P1#set mpls interface all
```

Results

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

```
interfaces {
    ge-0/0/1 {
```

```
        description P1-to-RR1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.6/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/2 {
        description P1-to-ABR1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.8/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/3 {
        description P1-to-PE1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.5/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/4 {
        description P1-to-ABR2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.10/31;
            }
```

```
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.2.2.8/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0008.00;
            }
        }
    }
}
policy-options {
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.0;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
}
routing-options {
    router-id 10.2.2.8;
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
    }
    ldp {
        interface all;
    }
    mpls {
```

```
        interface all;
    }
}
```

Configuring RR1 Device

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 RR1:

1. Configure the interfaces.

```
[edit interfaces]
user@RR1#set ge-0/0/1 description RR1-to-P1
user@RR1#set ge-0/0/1 vlan-tagging
user@RR1#set ge-0/0/1 unit 0 vlan-id 100
user@RR1#set ge-0/0/1 unit 0 family inet address 192.168.0.7/31
user@RR1#set ge-0/0/1 unit 0 family iso
user@RR1#set ge-0/0/1 unit 0 family mpls
```

2. Configure the loopback interface.

```
[edit interfaces]
user@RR1#set lo0 unit 1 family inet address 10.2.2.6/32
user@RR1#set lo0 unit 1 family iso address 49.0000.0000.aaaa.0006.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@RR1#set policy-statement add-noexport term 1 then community add noexport
user@RR1#set policy-statement allow-lo0 term 1 from interface lo0.1
user@RR1#set policy-statement allow-lo0 term 1 then accept
user@RR1#set policy-statement allow-lo0 term 2 then reject
user@RR1#set policy-statement export-inet3 term 1 from rib inet.3
user@RR1#set policy-statement export-inet3 term 1 then accept
user@RR1#set policy-statement export-inet3 term 2 then reject
```

```
user@RR1#set policy-statement pplb then load-balance per-packet
user@RR1#set community noexport members no-export
user@RR1#set community noexport members no-advertise
```

4. Configure routing options.

```
[edit routing-options]
user@RR1#set forwarding-table export pplb
user@RR1#set interface-routes rib-group inet inet0to3
user@RR1#set router-id 10.2.2.6
user@RR1#set autonomous-system 2
user@RR1#set rib-groups inet0to3 import-rib inet.0
user@RR1#set rib-groups inet0to3 import-rib inet.3
user@RR1#set rib-groups inet0to3 import-policy allow-lo0
user@RR1#set rib-groups inet3to0 import-rib inet.3
user@RR1#set rib-groups inet3to0 import-rib inet.0
user@RR1#set rib-groups inet3to0 import-rib inet6.3
user@RR1#set rib-groups inet3to0 import-policy add-noexport
```

5. Configure ISIS, RSVP, LDP, and MPLS protocols on the interface.

```
[edit protocols]
user@RR1#set isis level 1 disable
user@RR1#set isis interface all
user@RR1#set isis export allow-lo0
user@RR1#set isis topologies ipv6-unicast
user@RR1#set rsvp interface all
user@RR1#set ldp interface all
user@RR1#set mpls interface all
```

6. Configure BGP labeled unicast to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols bgp]
user@RR1#set path-selection external-router-id
user@RR1#set group toAs2Reg2BNs type internal
user@RR1#set group toAs2Reg2BNs family inet labeled-unicast rib-group inet3to0
user@RR1#set group toAs2Reg2BNs family inet labeled-unicast add-path receive
user@RR1#set group toAs2Reg2BNs family inet labeled-unicast add-path send path-count 4
user@RR1#set group toAs2Reg2BNs family inet labeled-unicast rib inet.3
user@RR1#set group toAs2Reg2BNs export export-inet3
```

```
user@RR1#set group toAs2Reg2BNs neighbor 10.2.2.3
user@RR1#set group toAs2Reg2BNs neighbor 10.2.2.4
user@RR1#set group toAs2Reg2BNs neighbor 10.2.2.5
user@RR1#set traceoptions file bgp.log
user@RR1#set traceoptions file size 100m
user@RR1#set traceoptions flag state detail
user@RR1#set traceoptions flag policy
user@RR1#set local-address 10.2.2.6
user@RR1#set cluster 10.2.2.6
```

Results

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

```
interfaces {
    ge-0/0/1 {
        description RR1-to-P1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.7/31;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 10.2.2.6/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0006.00;
            }
        }
    }
}
policy-options {
```

```
    policy-statement add-noexport {
        term 1 {
            then {
                community add noexport;
            }
        }
    }
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.1;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement export-inet3 {
        term 1 {
            from rib inet.3;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    community noexport members [ no-export no-advertise ];
}
routing-options {
    forwarding-table {
        export pplb;
    }
    interface-routes {
        rib-group inet inet0to3;
    }
    router-id 10.2.2.6;
    autonomous-system 2;
    rib-groups {
        inet0to3 {
```

```
            import-rib [ inet.0 inet.3 ];
            import-policy allow-lo0;
        }
        inet3to0 {
            import-rib [ inet.3 inet.0 inet6.3 ];
            import-policy add-noexport;
        }
    }
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
    }
    bgp {
        path-selection external-router-id;
        group toAs2Reg2BNs {
            type internal;
            family inet {
                labeled-unicast {
                    rib-group inet3to0;
                    add-path {
                        receive;
                        send {
                            path-count 4;
                        }
                    }
                    rib {
                        inet.3;
                    }
                }
            }
            export export-inet3;
            neighbor 10.2.2.3;
            neighbor 10.2.2.4;
            neighbor 10.2.2.5;
        }
        traceoptions {
```

```
            file bgp.log size 100m;
            flag state detail;
            flag policy;
        }
        local-address 10.2.2.6;
        cluster 10.2.2.6;
    }
    ldp {
        interface all;
    }
    mpls {
        interface all;
    }
}
```

Configuring ABR1 Device

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 ABR1:

1. Configure the interfaces to enable IP and MPLS transport.

```
[edit interfaces]
user@ABR1#set ge-0/0/1 description ABR1-to-P2
user@ABR1#set ge-0/0/1 vlan-tagging
user@ABR1#set ge-0/0/1 unit 0 vlan-id 100
user@ABR1#set ge-0/0/1 unit 0 family inet address 192.168.0.12/31
user@ABR1#set ge-0/0/1 unit 0 family iso
user@ABR1#set ge-0/0/1 unit 0 family mpls
user@ABR1#set ge-0/0/2 description ABR1-to-P1
user@ABR1#set ge-0/0/2 vlan-tagging
user@ABR1#set ge-0/0/2 unit 0 vlan-id 100
user@ABR1#set ge-0/0/2 unit 0 family inet address 192.168.0.9/31
user@ABR1#set ge-0/0/2 unit 0 family iso
user@ABR1#set ge-0/0/2 unit 0 family mpls
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@ABR1#set lo0 unit 0 family inet address 10.2.2.3/32
user@ABR1#set lo0 unit 0 family iso address 49.0000.0000.aaaa.0003.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@ABR1#set policy-statement allow-lo0 term 1 from interface lo0.0
user@ABR1#set policy-statement allow-lo0 term 1 then accept
user@ABR1#set policy-statement allow-lo0 term 2 then reject
user@ABR1#set policy-statement nhs term 1 from protocol bgp
user@ABR1#set policy-statement nhs term 1 then next-hop self
user@ABR1#set policy-statement nhs term 1 then accept
user@ABR1#set policy-statement pplb then load-balance per-packet
```

4. Apply per flow load balance policy to enable traffic protection.

```
[edit routing-options]
user@ABR1#set forwarding-table export pplb
user@ABR1#set router-id 10.2.2.3
user@ABR1#set autonomous-system 65002
```

5. Configure ISIS, RSVP, MPLS, and LDP protocols on the interface.

```
[edit protocols]
user@ABR1#set isis level 1 disable
user@ABR1#set isis interface all
user@ABR1#set isis export allow-lo0
user@ABR1#set isis topologies ipv6-unicast
user@ABR1#set rsvp interface all
user@ABR1#set ldp interface all
user@ABR1#set mpls label-switched-path toASBR2-gold to 10.2.2.2
user@ABR1#set mpls label-switched-path toASBR1-bronze to 10.2.2.1
user@ABR1#set mpls label-switched-path toASBR2-bronze to 10.2.2.2
user@ABR1#set mpls interface all
```

6. Configure BGP labeled unicast to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols]
user@ABR1#set bgp group toAs2RR type internal
user@ABR1#set bgp group toAs2RR local-address 10.2.2.3
user@ABR1#set bgp group toAs2RR advertise-inactive
user@ABR1#set bgp group toAs2RR family inet labeled-unicast add-path receive
user@ABR1#set bgp group toAs2RR family inet labeled-unicast add-path send path-count 4
user@ABR1#set bgp group toAs2RR family inet labeled-unicast rib inet.3
user@ABR1#set bgp group toAs2RR export nhs
user@ABR1#set bgp group toAs2RR cluster 10.2.2.3
user@ABR1#set bgp group toAs2RR neighbor 10.2.2.6
user@ABR1#set bgp group toAs2RR neighbor 10.2.2.7
user@ABR1#set bgp traceoptions file bgp.log
user@ABR1#set bgp traceoptions file size 100m
user@ABR1#set bgp traceoptions flag state detail
user@ABR1#set bgp traceoptions flag policy
```

Results

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

```
interfaces {
    ge-0/0/1 {
        description ABR1-to-P2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.12/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/2 {
        description ABR1-to-P1;
        vlan-tagging;
```

```
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.9/31;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.2.2.3/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0003.00;
            }
        }
    }
}
policy-options {
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.0;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement nhs {
        term 1 {
            from protocol bgp;
            then {
                next-hop self;
                accept;
            }
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
```

```
    }
}
routing-options {
    forwarding-table {
        export pplb;
    }
    router-id 10.2.2.3;
    autonomous-system 65002;
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
    }
    bgp {
        group toAs2RR {
            type internal;
            local-address 10.2.2.3;
            advertise-inactive;
            family inet {
                labeled-unicast {
                    add-path {
                        receive;
                        send {
                            path-count 4;
                        }
                    }
                    rib {
                        inet.3;
                    }
                }
            }
            export nhs;
            cluster 10.2.2.3;
            neighbor 10.2.2.6;
            neighbor 10.2.2.7;
        }
        traceoptions {
```

```
            file bgp.log size 100m;
            flag state detail;
            flag policy;
        }
    }
    ldp {
        interface all;
    }
    mpls {
        label-switched-path toASBR2-gold {
            to 10.2.2.2;
        }
        label-switched-path toASBR1-bronze {
            to 10.2.2.1;
        }
        label-switched-path toASBR2-bronze {
            to 10.2.2.2;
        }
        interface all;
    }
}
```

Configuring ABR2 Device

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 ABR2:

1. Configure the interfaces to enable IP and MPLS transport.

```
[edit interfaces]
user@ABR2#set ge-0/0/2 description ABR2-to-P2
user@ABR2#set ge-0/0/2 vlan-tagging
user@ABR2#set ge-0/0/2 unit 0 vlan-id 100
user@ABR2#set ge-0/0/2 unit 0 family inet address 192.168.0.14/31
user@ABR2#set ge-0/0/2 unit 0 family iso
user@ABR2#set ge-0/0/2 unit 0 family mpls
user@ABR2#set ge-0/0/4 description ABR2-to-P1
```

```
user@ABR2#set ge-0/0/4 vlan-tagging
user@ABR2#set ge-0/0/4 unit 0 vlan-id 100
user@ABR2#set ge-0/0/4 unit 0 family inet address 192.168.0.11/31
user@ABR2#set ge-0/0/4 unit 0 family iso
user@ABR2#set ge-0/0/4 unit 0 family mpls
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@ABR2#set lo0 unit 0 family inet address 2.2.2.4/32
user@ABR2#set lo0 unit 0 family iso address 49.0000.0000.aaaa.0004.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@ABR2#set policy-statement allow-lo0 term 1 from interface lo0.0
user@ABR2#set policy-statement allow-lo0 term 1 then accept
user@ABR2#set policy-statement allow-lo0 term 2 then reject
user@ABR2#set policy-statement nhs term 1 from protocol bgp
user@ABR2#set policy-statement nhs term 1 then next-hop self
user@ABR2#set policy-statement nhs term 1 then accept
user@ABR2#set policy-statement pplb then load-balance per-packet
```

4. Apply per flow load balance policy to enable traffic protection.

```
[edit routing-options]
user@ABR2#set forwarding-table export pplb
user@ABR2#set router-id 2.2.2.4
user@ABR2#set autonomous-system 2
```

5. Configure ISIS, RSVP, MPLS, and LDP protocols on the interface.

```
[edit protocols]
user@ABR2#set isis level 1 disable
user@ABR2#set isis interface all
user@ABR2#set isis export allow-lo0
user@ABR2#set isis topologies ipv6-unicast
user@ABR2#set rsvp interface all
```

```
user@ABR2#set ldp interface all
user@ABR2#set mpls label-switched-path toASBR1-bronze to 2.2.2.1
user@ABR2#set mpls interface all
```

6. Configure BGP labeled unicast to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols]
user@ABR2#set bgp group toAs2RR type internal
user@ABR2#set bgp group toAs2RR local-address 2.2.2.4
user@ABR2#set bgp group toAs2RR advertise-inactive
user@ABR2#set bgp group toAs2RR family inet labeled-unicast add-path receive
user@ABR2#set bgp group toAs2RR family inet labeled-unicast add-path send path-count 4
user@ABR2#set bgp group toAs2RR family inet labeled-unicast rib inet.3
user@ABR2#set bgp group toAs2RR export nhs
user@ABR2#set bgp group toAs2RR cluster 2.2.2.4
user@ABR2#set bgp group toAs2RR neighbor 2.2.2.6
user@ABR2#set bgp group toAs2RR neighbor 2.2.2.7
user@ABR2#set bgp traceoptions file bgp.log
user@ABR2#set bgp traceoptions file size 100m
user@ABR2#set bgp traceoptions flag state detail
user@ABR2#set bgp traceoptions flag policy
```

Results

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

```
interfaces {
    ge-0/0/2 {
        description ABR2-to-P2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.14/31;
            }
            family iso;
            family mpls;
        }
```

```
    }
    ge-0/0/4 {
        description ABR2-to-P1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.11/31;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 2.2.2.4/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0004.00;
            }
        }
    }
}
policy-options {
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.0;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement nhs {
        term 1 {
            from protocol bgp;
            then {
                next-hop self;
                accept;
            }
        }
    }
```

```
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
routing-options {
    forwarding-table {
        export pplb;
    }
    router-id 2.2.2.4;
    autonomous-system 2;
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
    }
    bgp {
        group toAs2RR {
            type internal;
            local-address 2.2.2.4;
            advertise-inactive;
            family inet {
                labeled-unicast {
                    add-path {
                        receive;
                        send {
                            path-count 4;
                        }
                    }
                    rib {
                        inet.3;
                    }
                }
            }
            export nhs;
            cluster 2.2.2.4;
```

```
            neighbor 2.2.2.6;
            neighbor 2.2.2.7;
        }
        traceoptions {
            file bgp.log size 100m;
            flag state detail;
            flag policy;
        }
    }
    ldp {
        interface all;
    }
    mpls {
        label-switched-path toASBR1-bronze {
            to 2.2.2.1;
        }
        interface all;
    }
}
```

Configuring P2 Device

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 P2:

1. Configure the interfaces to enable IP and MPLS transport.

```
[edit interfaces]
user@P2#set ge-0/0/1 description P2-to-ABR1
user@P2#set ge-0/0/1 vlan-tagging
user@P2#set ge-0/0/1 unit 0 vlan-id 100
user@P2#set ge-0/0/1 unit 0 family inet address 192.168.0.13/31
user@P2#set ge-0/0/1 unit 0 family iso
user@P2#set ge-0/0/1 unit 0 family mpls
user@P2#set ge-0/0/2 description P2-to-ABR2
user@P2#set ge-0/0/2 vlan-tagging
user@P2#set ge-0/0/2 unit 0 vlan-id 100
```

```
user@P2#set ge-0/0/2 unit 0 family inet address 192.168.0.15/31
user@P2#set ge-0/0/2 unit 0 family iso
user@P2#set ge-0/0/2 unit 0 family mpls
user@P2#set ge-0/0/3 description P2-to-RR2
user@P2#set ge-0/0/3 vlan-tagging
user@P2#set ge-0/0/3 unit 0 vlan-id 100
user@P2#set ge-0/0/3 unit 0 family inet address 192.168.0.16/31
user@P2#set ge-0/0/3 unit 0 family iso
user@P2#set ge-0/0/3 unit 0 family mpls
user@P2#set ge-0/0/4 description P2-to-ASBR1
user@P2#set ge-0/0/4 vlan-tagging
user@P2#set ge-0/0/4 unit 0 vlan-id 100
user@P2#set ge-0/0/4 unit 0 family inet address 192.168.0.18/31
user@P2#set ge-0/0/4 unit 0 family iso
user@P2#set ge-0/0/4 unit 0 family mpls
user@P2#set ge-0/0/5 description P2-to-ASBR2
user@P2#set ge-0/0/5 vlan-tagging
user@P2#set ge-0/0/5 unit 0 vlan-id 100
user@P2#set ge-0/0/5 unit 0 family inet address 192.168.0.20/31
user@P2#set ge-0/0/5 unit 0 family iso
user@P2#set ge-0/0/5 unit 0 family mpls
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@P2#set lo0 unit 0 family inet address 2.2.2.9/32
user@P2#set lo0 unit 0 family iso address 49.0000.0000.aaaa.0009.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@P2#set policy-statement allow-lo0 term 1 from interface lo0.0
user@P2#set policy-statement allow-lo0 term 1 then accept
user@P2#set policy-statement allow-lo0 term 2 then reject
```

4. Configure routing options.

```
[edit routing-options]
user@P2#set router-id 2.2.2.9
```

5. Configure ISIS, RSVP, MPLS, and LDP protocols on the interface.

```
[edit protocols]
user@P2#set isis level 1 disable
user@P2#set isis interface all
user@P2#set isis export allow-lo0
user@P2#set isis topologies ipv6-unicast
user@P2#set rsvp interface all
user@P2#set ldp interface all
user@P2#set mpls interface all
```

Results

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

```
interfaces {
    ge-0/0/1 {
        description P2-to-ABR1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.13/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/2 {
        description P2-to-ABR2;
        vlan-tagging;
        unit 0 {
```

```
            vlan-id 100;
            family inet {
                address 192.168.0.15/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/3 {
        description P2-to-RR2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.16/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/4 {
        description P2-to-ASBR1;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.18/31;
            }
            family iso;
            family mpls;
        }
    }
    ge-0/0/5 {
        description P2-to-ASBR2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.20/31;
            }
            family iso;
            family mpls;
        }
```

```
    }
    lo0 {
        unit 0 {
            family inet {
                address 2.2.2.9/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0009.00;
            }
        }
    }
}
policy-options {
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.0;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
}
routing-options {
    router-id 2.2.2.9;
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
    }
    ldp {
        interface all;
    }
    mpls {
        interface all;
```

```
    }
}
```

Configuring RR2 Device

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 RR2:

1. Configure the interfaces to enable IP and MPLS transport.

```
[edit interfaces]
user@RR2#set ge-0/0/3 description RR2-to-P2
user@RR2#set ge-0/0/3 vlan-tagging
user@RR2#set ge-0/0/3 unit 0 vlan-id 100
user@RR2#set ge-0/0/3 unit 0 family inet address 192.168.0.17/31
user@RR2#set ge-0/0/3 unit 0 family iso
user@RR2#set ge-0/0/3 unit 0 family mpls
```

2. Configure the loopback interface to be used as router ID and termination interface for LDP and BGP sessions.

```
[edit interfaces]
user@RR2#set lo0 unit 1 family inet address 2.2.2.7/32
user@RR2#set lo0 unit 1 family iso address 49.0000.0000.aaaa.0007.00
```

3. Configure multipath resolution policies to install hierarchical multipaths into PFE.

```
[edit policy-options]
user@RR2#set policy-statement allow-lo0 term 1 from interface lo0.1
user@RR2#set policy-statement allow-lo0 term 1 then accept
user@RR2#set policy-statement allow-lo0 term 2 then reject
user@RR2#set policy-statement export-inet3 term 1 from rib inet.3
user@RR2#set policy-statement export-inet3 term 1 then accept
```

```
user@RR2#set policy-statement export-inet3 term 2 then reject
user@RR2#set policy-statement pplb then load-balance per-packet
```

4. Apply per flow load balance policy to enable traffic protection.

```
[edit routing-options]
user@RR2#set forwarding-table export pplb
user@RR2#set interface-routes rib-group inet inet0to3
user@RR2#set router-id 2.2.2.7
user@RR2#set autonomous-system 2
user@RR2#set rib-groups inet0to3 import-rib inet.0
user@RR2#set rib-groups inet0to3 import-rib inet.3
user@RR2#set rib-groups inet0to3 import-policy allow-lo0
```

5. Configure ISIS, RSVP, MPLS, and LDP protocols on the interface.

```
[edit protocols]
user@RR2#set isis level 1 disable
user@RR2#set isis interface all
user@RR2#set isis export allow-lo0
user@RR2#set isis topologies ipv6-unicast
user@RR2#set rsvp interface all
user@RR2#set ldp interface all
user@RR2#set mpls interface all
```

6. Configure BGP labeled unicast to exchange loopback IP addresses as BGP labeled unicast prefixes.

```
[edit protocols]
user@RR2#set bgp path-selection external-router-id
user@RR2#set bgp group toAs2Reg1BNs type internal
user@RR2#set bgp group toAs2Reg1BNs family inet labeled-unicast add-path receive
user@RR2#set bgp group toAs2Reg1BNs family inet labeled-unicast add-path send path-count 4
user@RR2#set bgp group toAs2Reg1BNs family inet labeled-unicast rib inet.3
user@RR2#set bgp group toAs2Reg1BNs neighbor 2.2.2.1
user@RR2#set bgp group toAs2Reg1BNs neighbor 2.2.2.2
user@RR2#set bgp group toAs2Reg1BNs neighbor 2.2.2.3
user@RR2#set bgp group toAs2Reg1BNs neighbor 2.2.2.4
user@RR2#set bgp traceoptions file bgp.log
user@RR2#set bgp traceoptions file size 100m
user@RR2#set bgp traceoptions flag state detail
```

```
user@RR2#set bgp traceoptions flag policy
user@RR2#set bgp local-address 2.2.2.7
user@RR2#set bgp cluster 2.2.2.7
```

Results

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

```
interfaces {
    ge-0/0/3 {
        description RR2-to-P2;
        vlan-tagging;
        unit 0 {
            vlan-id 100;
            family inet {
                address 192.168.0.17/31;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 2.2.2.7/32;
            }
            family iso {
                address 49.0000.0000.aaaa.0007.00;
            }
        }
    }
}
policy-options {
    policy-statement allow-lo0 {
        term 1 {
            from interface lo0.1;
            then accept;
        }
        term 2 {
```

```
            then reject;
        }
    }
    policy-statement export-inet3 {
        term 1 {
            from rib inet.3;
            then accept;
        }
        term 2 {
            then reject;
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
routing-options {
    forwarding-table {
        export pplb;
    }
    interface-routes {
        rib-group inet inet0to3;
    }
    router-id 2.2.2.7;
    autonomous-system 2;
    rib-groups {
        inet0to3 {
            import-rib [ inet.0 inet.3 ];
            import-policy allow-lo0;
        }
    }
}
protocols {
    isis {
        level 1 disable;
        interface all;
        export allow-lo0;
        topologies ipv6-unicast;
    }
    rsvp {
        interface all;
```

```
    }
    bgp {
        path-selection external-router-id;
        group toAs2Reg1BNs {
            type internal;
            family inet {
                labeled-unicast {
                    add-path {
                        receive;
                        send {
                            path-count 4;
                        }
                    }
                    rib {
                        inet.3;
                    }
                }
            }
            neighbor 2.2.2.1;
            neighbor 2.2.2.2;
            neighbor 2.2.2.3;
            neighbor 2.2.2.4;
        }
        traceoptions {
            file bgp.log size 100m;
            flag state detail;
            flag policy;
        }
        local-address 2.2.2.7;
        cluster 2.2.2.7;
    }
    ldp {
        interface all;
    }
    mpls {
        interface all;
    }
}
```

Verification

IN THIS SECTION

- Verifying that Nexthops are Resolved | 819
- Verifying the Nexthop Entries in the Routing Table | 820

Confirm that the configuration is working properly.

Verifying that Nexthops are Resolved

Purpose

Verify that PE2 and PE3 nexthops are resolved at PE1.

Action

From operational mode, run the `show route forwarding-table destination` command.

```
user@PE1> show route forwarding-table destination 10.3.3.3 extensive table default | match Weight

                                  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x4000
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x1
```

```
  Next-hop interface: ge-0/0/3.0    Weight: 0x1
```

```
user@PE1> show route forwarding-table destination 10.3.3.3 extensive table red | match Weight

                                  Weight: 0x1
                                  Weight: 0x1
  Next-hop interface: ge-0/0/3.0  Weight: 0x1
                                  Weight: 0x4000
                                  Weight: 0x4000
  Next-hop interface: ge-0/0/3.0  Weight: 0x4000
```

Meaning

You can see weights `0x1` and `0x4000` for primary and backup nexthops.

Verifying the Nexthop Entries in the Routing Table

Purpose

Verify the active nexthop routing entries at PE1.

Action

From operational mode, run the `show route extensive expanded-nh` command.

```
user@PE1> show route 10.3.3.3 extensive expanded-nh

inet.0: 36 destinations, 65 routes (36 active, 0 holddown, 0 hidden)
10.3.3.3/32 (2 entries, 1 announced)
Installed-nexthop:
List (0xd6ba4b8) Index:1048626
  Indr (0xc593cac) 10.1.1.1
    Krt_inh (0xcc14684) Index:1048614
      List (0xc4cf7b4) Index:1048613
        Frr_inh (0xc592730) Index:1048608
          Chain (0xc59334c) Index:651 Push 300368
            Router (0xc58ea40) Index:628 192.168.0.5 Push 299808
```

```
        Frr_inh (0xc592604) Index:1048609
          Chain (0xc5924d8) Index:649 Push 300384
            Router (0xc58ea40) Index:628 192.168.0.5 Push 299808
        Frr_inh (0xc592154) Index:1048611
          Chain (0xc591bdc) Index:654 Push 300368
            Router (0xc58ebd0) Index:629 192.168.0.5 Push 299824
        Frr_inh (0xc5921b8) Index:1048612
          Chain (0xc591a4c) Index:655 Push 300384
            Router (0xc58ebd0) Index:629 192.168.0.5 Push 299824
  Indr (0xc593ab8) 10.1.1.2
    Krt_inh (0xcc14f84) Index:1048624
      List (0xc4d0074) Index:1048623
        Frr_inh (0xc5939f0) Index:1048619
          Chain (0xc592ab4) Index:638 Push 300144
            Router (0xc58ea40) Index:628 192.168.0.5 Push 299808
        Frr_inh (0xc593a54) Index:1048620
          Chain (0xc591efc) Index:637 Push 300160
            Router (0xc58ea40) Index:628 192.168.0.5 Push 299808
        Frr_inh (0xc59172c) Index:1048589
          Chain (0xc5903a4) Index:640 Push 300144
            Router (0xc58ebd0) Index:629 192.168.0.5 Push 299824
        Frr_inh (0xc59159c) Index:1048590
          Chain (0xc58fa44) Index:639 Push 300160
            Router (0xc58ebd0) Index:629 192.168.0.5 Push 299824
TSI:
<SNIP>
               Protocol next hop: 10.1.1.1
                Indirect next hop: 0xcc14684 1048614 INH Session ID: 0x146 Weight 0x1
                Protocol next hop: 10.1.1.2
                Indirect next hop: 0xcc14f84 1048624 INH Session ID: 0x145 Weight 0x4000
                State: >Active Ext>
                Local AS:     65002 Peer AS:     65001
<SNIP>
                Indirect next hops: 2
                        Protocol next hop: 10.1.1.1 Metric: 1
                        Indirect next hop: 0xcc14684 1048614 INH Session ID: 0x146 Weight 0x1
                        Indirect path forwarding next hops (Merged): 4
<SNIP>
                        Protocol next hop: 10.1.1.2 Metric: 1
                        Indirect next hop: 0xcc14f84 1048624 INH Session ID: 0x145 Weight 0x4000
                        Indirect path forwarding next hops (Merged): 4
```

Meaning

You can see the weights `0x1` and `0x4000` for primary and backup nexthops.

FAT Pseudowire Support for BGP L2VPN and VPLS Overview

A pseudowire is a Layer 2 circuit or service that emulates the essential attributes of a telecommunications service, such as a T1 line, over an MPLS packet-switched network (PSN). The pseudowire is intended to provide only the minimum necessary functionality to emulate the wire with the required resiliency requirements for the given service definition.

In an MPLS network, the flow-aware transport (FAT) of pseudowires flow label, as described in *draft-keyupdate-l2vpn-fat-pw-bgp*, is used for load-balancing traffic across BGP-signaled pseudowires for the Layer 2 virtual private network (L2VPN) and virtual private LAN service (VPLS).

FAT flow label is configured only on the label edge routers (LERs). This causes the transit routers or label-switching routers (LSRs) to perform load balancing of MPLS packets across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs) without the need for deep packet inspection of the payload.

FAT flow label can be used for LDP-signaled forwarding equivalence class (FEC 128 and FEC 129) pseudowires for VPWS and VPLS pseudowires. The interface parameter (Sub-TLV) is used both for FEC 128 and FEC 129 pseudowires. The sub-TLV defined for LDP contains the transmit (T) and receive (R) bits. The T bit advertises the ability to push the flow label. The R bit advertises the ability to pop the flow label. By default, the signaling behavior of the provider edge (PE) router for any of these pseudowires is to advertise the T and R bits in the label set to 0.

The `flow-label-transmit` and `flow-label-receive` configuration statements provide the ability to set the T bit and R bit advertisement to 1 in the Sub-TLV field, which is part of the interface parameters of the FEC for the LDP label-mapping message. You can use these statements to control the pushing of the load-balancing label and the advertisement of the label to the routing peers in the control plane for BGP signaled pseudowires like L2VPN and VPLS.

SEE ALSO

Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **859**

Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **860**

Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **824**

flow-label-receive

flow-label-transmit

Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic

The flow-aware transport (FAT) or flow label is supported for BGP-signaled pseudowires such as L2VPN to be configured only on the label edge routers (LERs). This enables the transit routers or the label-switching routers (LSRs) to perform load balancing of MPLS packets across equal-cost multipath paths (ECMP) or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. FAT pseudowires or flow label can be used with LDP-signaled L2VPNs with forwarding equivalence class (FEC128 and FEC129), and the support for flow label is extended for BGP-signaled pseudowires for point-to-point or point–to-multipoint Layer 2 services.

Before you configure FAT pseudowire support for BGP L2VPN to load-balance MPLS traffic:

- Configure the device interfaces and enable MPLS on all the interfaces.
- Configure RSVP.
- Configure MPLS and an LSP to the remote PE router.
- Configure BGP and OSPF.

To configure FAT pseudowire support for BGP L2VPN to load-balance MPLS traffic, you must do the following:

1. Configure the sites connected to the provider equipment for a given routing instance for the L2VPN protocols.

```
[edit routing-instances routing-instance name protocols l2vpn]
user@host#  set site site-name site-identifier site-identifier
user@host#  set site site-name interface interface-name remote-site-id remote-site-id
```

2. Configure the L2VPN protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE.

```
[edit routing-instances routing-instance name protocols l2vpn]
user@host#  set flow-label-receive
```

3. Configure the L2VPN protocol to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances routing-instance name protocols l2vpn]
user@host#  set flow-label-transmit
```

4. Configure the sites connected to the provider equipment for a given routing instance for the VPLS protocol.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set site site-name site-identifier site-identifier
user@host#  set site-range site-range
```

5. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set flow-label-receive
```

6. Configure the VPLS protocol to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set flow-label-transmit
```

SEE ALSO

FAT Pseudowire Support for BGP L2VPN and VPLS Overview | 822

Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | 859

Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | 860

flow-label-receive

flow-label-transmit

Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic

IN THIS SECTION

Requirements | 825

- Overview | **825**
- Configuration | **826**
- Configuring PE2 | **844**
- Verification | **852**

This example shows how to implement FAT pseudowire support for BGP L2VPN to help load-balance MPLS traffic.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers
- Junos OS Release 16.1 or later running on all devices

Before you configure FAT pseudowire support for BGP L2VPN, be sure you configure the routing and signaling protocols.

Overview

IN THIS SECTION

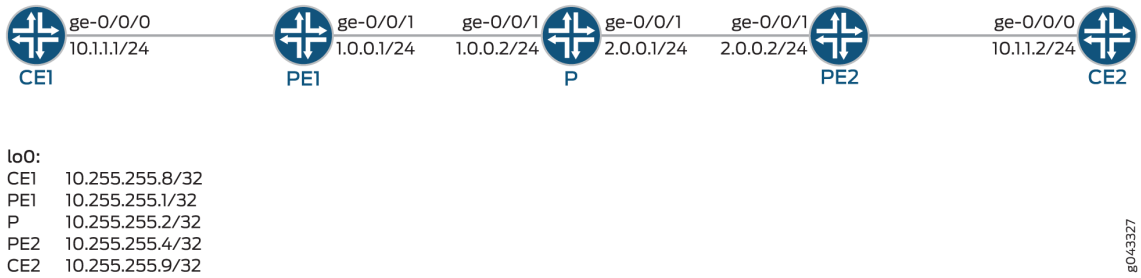
- Topology | **825**

Junos OS allows the flow-aware transport (FAT) flow label that is supported for BGP-signaled pseudowires such as L2VPN to be configured only on the label edge routers (LERs). This causes the transit routers or the label-switching routers(LSRs) to perform load balancing of MPLS packets across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. The FAT flow label can be used for LDP-signaled forwarding equivalence class (FEC 128 and FEC 129) pseudowires for VPWS and VPLS pseudowires.

Topology

Figure 56 on page 826, shows the FAT pseudowire support for BGP L2VPN configured on Device PE1 and Device PE2.

Figure 56: Example FAT Pseudowire Support for BGP L2VPN



Configuration

IN THIS SECTION

- Verification | 837

CLI Quick Configuration

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

CE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family inet address 10.1.1.1/24
set interfaces lo0 unit 0 family inet address 10.255.255.8/32
```

PE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mtu 1600
set interfaces ge-0/0/0 encapsulation vlan-ccc
set interfaces ge-0/0/0 unit 300 encapsulation vlan-ccc
set interfaces ge-0/0/0 unit 300 vlan-id 600
set interfaces ge-0/0/0 unit 600 encapsulation vlan-vpls
```

```
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family vpls
set interfaces ge-0/0/1 unit 0 family inet address 1.0.0.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.1/32
set routing-options nonstop-routing
set routing-options router-id 10.255.255.1
set routing-options autonomous-system 100
set routing-options forwarding-table export exp-to-frwd
set protocols rsvp interface all
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols mpls label-switched-path to-pe2 to 10.255.255.4
set protocols mpls interface ge-0/0/1.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.1
set protocols bgp group vpls-pe family l2vpn auto-discovery-only
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.4
set protocols bgp group vpls-pe neighbor 10.255.255.2
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
set policy-options policy-statement exp-to-frwd term 0 from community vpls-com
set policy-options policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe2
set policy-options policy-statement exp-to-frwd term 0 then accept
set policy-options community vpls-com members target:100:100
set routing-instances l2vpn-inst instance-type l2vpn
set routing-instances l2vpn-inst interface ge-0/0/0.300
set routing-instances l2vpn-inst route-distinguisher 10.255.255.1:200
set routing-instances l2vpn-inst vrf-target target:100:100
set routing-instances l2vpn-inst protocols l2vpn encapsulation-type ethernet-vlan
set routing-instances l2vpn-inst protocols l2vpn site pe1 site-identifier 1
set routing-instances l2vpn-inst protocols l2vpn site pe1 interface ge-0/0/0.300 remote-site-id 2
set routing-instances l2vpn-inst protocols l2vpn flow-label-transmit
set routing-instances l2vpn-inst protocols l2vpn flow-label-receive
set routing-instances vpl1 instance-type vpls
set routing-instances vpl1 interface ge-0/0/0.600
set routing-instances vpl1 route-distinguisher 10.255.255.1:100
set routing-instances vpl1 vrf-target target:100:100
set routing-instances vpl1 protocols vpls site-range 10
set routing-instances vpl1 protocols vpls no-tunnel-services
set routing-instances vpl1 protocols vpls site vpl1PE1 site-identifier 1
```

```
set routing-instances vpl1 protocols vpls flow-label-transmit
set routing-instances vpl1 protocols vpls flow-label-receive
```

P

```
set interfaces ge-0/0/0 unit 0 family inet address 1.0.0.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 2.0.0.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 100
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.2
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.1
set protocols bgp group vpls-pe neighbor 10.255.255.4 deactivate protocols bgp
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
```

PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 2.0.0.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 encapsulation vlan-ccc
set interfaces ge-0/0/1 unit 300 encapsulation vlan-ccc
set interfaces ge-0/0/1 unit 300 vlan-id 600
set interfaces ge-0/0/1 unit 600 encapsulation vlan-vpls
set interfaces ge-0/0/1 unit 600 vlan-id 600
set interfaces ge-0/0/1 unit 600 family vpls
set interfaces lo0 unit 0 family inet address 10.255.255.4/32
set routing-options router-id 10.255.255.4
```

```
set routing-options autonomous-system 100
set routing-options forwarding-table export exp-to-frwd
set protocols rsvp interface all
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols mpls label-switched-path to-pe1 to 10.255.255.1
set protocols mpls interface ge-0/0/0.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.4
set protocols bgp group vpls-pe family l2vpn auto-discovery-only
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.1
set protocols bgp group vpls-pe neighbor 10.255.255.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set policy-options policy-statement exp-to-frwd term 0 from community vpls-com
set policy-options policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe1
set policy-options policy-statement exp-to-frwd term 0 then accept
set policy-options community vpls-com members target:100:100
set routing-instances l2vpn-inst instance-type l2vpn
set routing-instances l2vpn-inst interface ge-0/0/1.300
set routing-instances l2vpn-inst route-distinguisher 10.255.255.4:200
set routing-instances l2vpn-inst vrf-target target:100:100
set routing-instances l2vpn-inst protocols l2vpn encapsulation-type ethernet-vlan
set routing-instances l2vpn-inst protocols l2vpn site pe2 site-identifier 2
set routing-instances l2vpn-inst protocols l2vpn site pe2 interface ge-0/0/1.300 remote-site-id 1
set routing-instances l2vpn-inst protocols l2vpn flow-label-transmit
set routing-instances l2vpn-inst protocols l2vpn flow-label-receive
set routing-instances vpl1 instance-type vpls
set routing-instances vpl1 interface ge-0/0/1.600
set routing-instances vpl1 route-distinguisher 10.255.255.4:100
set routing-instances vpl1 vrf-target target:100:100
set routing-instances vpl1 protocols vpls site-range 10
set routing-instances vpl1 protocols vpls no-tunnel-services
set routing-instances vpl1 protocols vpls site vpl1PE2 site-identifier 2
set routing-instances vpl1 protocols vpls flow-label-transmit
set routing-instances vpl1 protocols vpls flow-label-receive
deactivate routing-instances vpl1
```

CE2

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family inet address 10.1.1.2/24
set interfaces lo0 unit 0 family inet address 10.255.255.9/32
```

Configuring PE1

Step-by-Step Procedure

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

To configure Device PE1:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-0/0/0 vlan-tagging
user@PE1# set ge-0/0/0 mtu 1600
user@PE1# set ge-0/0/0 encapsulation vlan-ccc
user@PE1# set ge-0/0/0 unit 300 encapsulation vlan-ccc
user@PE1# set ge-0/0/0 unit 300 vlan-id 600
user@PE1# set ge-0/0/0 unit 600 encapsulation vlan-vpls
user@PE1# set ge-0/0/0 unit 600 vlan-id 600
user@PE1# set ge-0/0/0 unit 600 family vpls deactivate interfaces ge-0/0/0 unit 600
user@PE1# set ge-0/0/1 unit 0 family inet address 1.0.0.1/24
user@PE1# set ge-0/0/1 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.255.255.1/32
```

2. Configure nonstop routing, and configure the router ID.

```
[edit routing-options]
user@PE1# set nonstop-routing
user@PE1# set router-id 10.255.255.1
```

3. Configure the autonomous system (AS) number, and apply the policy to the forwarding table of the local router with the export statement.

```
[edit routing-options]
user@PE1# set autonomous-system 100
user@PE1# set forwarding-table export exp-to-frwd
```

4. Configure the RSVP protocol on the interfaces.

```
[edit protocols rsvp]
user@PE1# set interface all
user@PE1# set interface ge-0/0/1.0
user@PE1# set interface lo0.0
```

5. Apply the label-switched path attributes to the MPLS protocol, and configure the interface.

```
[edit protocols mpls]
user@PE1# set label-switched-path to-pe2 to 10.255.255.4
user@PE1# set interface ge-0/0/1.0
```

6. Define a peer group, and configure the address of the local-end address of the BGP session for peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE1# set type internal
user@PE1# set local-address 10.255.255.1
```

7. Configure attributes of the protocol family for NLRIs in updates.

```
[edit protocols bgp group vpls-pe]
user@PE1# set family l2vpn auto-discovery-only
user@PE1# set family l2vpn signaling
```

8. Configure neighbors for the peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.2
```

9. Configure traffic engineering, and configure the interfaces of OSPF area 0.0.0.0.

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

10. Configure the routing policy and the BGP community information.

```
[edit policy-options]
user@PE1# set policy-statement exp-to-fwd term 0 from community vpls-com
user@PE1# set policy-statement exp-to-fwd term 0 then install-nexthop lsp to-pe2
user@PE1# set policy-statement exp-to-fwd term 0 then accept
user@PE1# set community vpls-com members target:100:100
```

11. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances l2vpn-inst]
user@PE1# set instance-type l2vpn
user@PE1# set interface ge-0/0/0.300
```

12. Configure the route distinguisher for instance `l2vpn-inst`, and configure the VRF target community.

```
[edit routing-instances l2vpn-inst]
user@PE1# set route-distinguisher 10.255.255.1:200
user@PE1# set vrf-target target:100:100
```

13. Configure the type of encapsulation required for the L2VPN protocol.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE1# set encapsulation-type ethernet-vlan
```

14. Configure the sites connected to the provider equipment.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE1# set site pe1 site-identifier 1
user@PE1# set site pe1 interface ge-0/0/0.300 remote-site-id 2
```

15. Configure the L2VPN protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE1# set flow-label-transmit
user@PE1# set flow-label-receive
```

16. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances vpl1]
user@PE1# set instance-type vpls
user@PE1# set interface ge-0/0/0.600
```

17. Configure the route distinguisher for instance `vp1`, and configure the VRF target community.

```
[edit routing-instances vpl1]
user@PE1# set route-distinguisher 10.255.255.1:100
user@PE1# set vrf-target target:100:100
```

18. Assign the maximum site identifier for the VPLS domain.

```
[edit routing-instances vpl1 protocols vpls]
user@PE1# set site-range 10
```

19. Configure to not use the tunnel services for the VPLS instance, and assign a site identifier to the site connected to the provider equipment.

```
[edit routing-instances vpl1 protocols vpls]
user@PE1# set no-tunnel-services
user@PE1# set site vpl1PE1 site-identifier 1
```

20. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances vpl1 protocols vpls]
user@PE1# set flow-label-transmit
user@PE1# set flow-label-receive
```

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-0/0/0 {
    vlan-tagging;
    mtu 1600;
    encapsulation vlan-ccc;
    unit 300 {
        encapsulation vlan-ccc;
        vlan-id 600;
    }
     unit 600 {
        encapsulation vlan-vpls;
        vlan-id 600;
        family vpls;
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
```

```
            address 1.0.0.1/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.1/32;
        }
    }
}
```

```
user@PE1# show protocols
rsvp {
    interface all;
    interface ge-0/0/1.0;
    interface lo0.0;
}
mpls {
    label-switched-path to-pe2 {
        to 10.255.255.4;
    }
    interface ge-0/0/1.0;
}
bgp {
    group vpls-pe {
        type internal;
        local-address 10.255.255.1;
        family l2vpn {
            auto-discovery-only;
            signaling;
        }
        neighbor 10.255.255.4;
        neighbor 10.255.255.2;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
```

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

```
user@PE1# show policy-options
policy-statement exp-to-frwd {
    term 0 {
        from community vpls-com;
        then {
            install-nexthop lsp to-pe2;
            accept;
        }
    }
}
community vpls-com members target:100:100;
```

```
user@PE1# show routing-instances
l2vpn-inst {
    instance-type l2vpn;
    interface ge-0/0/0.300;
    route-distinguisher 10.255.255.1:200;
    vrf-target target:100:100;
    protocols {
        l2vpn {
            encapsulation-type ethernet-vlan;
            site pe1 {
                site-identifier 1;
                interface ge-0/0/0.300 {
                    remote-site-id 2;
                }
            }
            flow-label-transmit;
            flow-label-receive;
        }
    }
}
 vpl1 {
    instance-type vpls;
```

```
    interface ge-0/0/0.600;
    route-distinguisher 10.255.255.1:100;
    vrf-target target:100:100;
    protocols {
        vpls {
            site-range 10;
            no-tunnel-services;
            site vpl1PE1 {
                site-identifier 1;
            }
            flow-label-transmit;
            flow-label-receive;
        }
    }
}
```

```
user@PE1# show routing-options
nonstop-routing;
router-id 10.255.255.1;
autonomous-system 100;
forwarding-table {
    export exp-to-frwd;
}
```

Verification

IN THIS SECTION

- Verifying the BGP Summary Information | 838
- Verifying the L2VPN Connections Information | 838
- Verifying the Routes | 840

Confirm that the configuration is working properly.

Verifying the BGP Summary Information

Purpose

Verify the BGP summary information.

Action

From operational mode, enter the `show bgp summary` command.

```
user@PE1> show bgp summary

Groups: 1 Peers: 2 Down peers: 1
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l2vpn.0
                       1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.255.255.2            100          0          0       0       0 2d 12:54:28 Active
10.255.255.4            100       8121       8093       0       0 2d 12:53:56 Establ
  bgp.l2vpn.0: 1/1/1/0
  l2vpn-inst.l2vpn.0: 1/1/1/0
```

Meaning

The output displays the BGP summary information.

Verifying the L2VPN Connections Information

Purpose

Verify the Layer 2 VPN connections information.

Action

From operational mode, run the `show l2vpn connections` command to display the Layer 2 VPN connections information.

```
user@PE1> show l2vpn connections
```

```
Layer-2 VPN connections:

Legend for connection status (St)
EI -- encapsulation invalid      NC -- interface encapsulation not CCC/TCC/VPLS
EM -- encapsulation mismatch     WE -- interface and instance encaps not same
VC-Dn -- Virtual circuit down    NP -- interface hardware not present
CM -- control-word mismatch      -> -- only outbound connection is up
CN -- circuit not provisioned    <- -- only inbound connection is up
OR -- out of range               Up -- operational
OL -- no outgoing label          Dn -- down
LD -- local site signaled down   CF -- call admission control failure
RD -- remote site signaled down  SC -- local and remote site ID collision
LN -- local site not designated  LM -- local site ID not minimum designated
RN -- remote site not designated RM -- remote site ID not minimum designated
XX -- unknown connection status  IL -- no incoming label
MM -- MTU mismatch               MI -- Mesh-Group ID not available
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy
RS -- remote site standby        SN -- Static Neighbor
LB -- Local site not best-site   RB -- Remote site not best-site
VM -- VLAN ID mismatch

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

Instance: l2vpn-inst
Edge protection: Not-Primary
  Local site: pe1 (1)
    connection-site           Type  St     Time last up          # Up trans
    2                         rmt   Up     Jun 22 14:46:50 2015           1
      Remote PE: 10.255.255.4, Negotiated control-word: Yes (Null)
      Incoming label: 800003, Outgoing label: 800002
      Local interface: ge-0/0/0.300, Status: Up, Encapsulation: VLAN
      Flow Label Transmit: Yes, Flow Label Receive: Yes
```

Meaning

The output displays the Layer 2 VPN connections information along with the flow label transmit and flow label receive information.

Verifying the Routes

Purpose

Verify that the expected routes are learned.

Action

From operational mode, run the `show route` command to display the routes in the routing table.

```
user@PE1> show route
inet.0: 51 destinations, 51 routes (51 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.0.0.0/24         *[Direct/0] 2d 12:48:34
                    > via ge-0/0/1.0
1.0.0.1/32         *[Local/0] 2d 12:48:34
                      Local via ge-0/0/1.0
2.0.0.0/24         *[OSPF/10] 2d 12:48:24, metric 2
                    > to 1.0.0.2 via ge-0/0/1.0
10.4.0.0/16        *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.5.0.0/16        *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.6.128.0/17      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.9.0.0/16        *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.10.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.13.4.0/23       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.13.10.0/23      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.82.0.0/15       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.84.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.85.12.0/22      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.92.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
```

```
10.94.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.99.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.102.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.102.160.0/19    *[Direct/0] 2d 12:48:34
                    > via fxp0.0
10.102.169.99/32   *[Local/0] 2d 12:48:34
                      Local via fxp0.0
10.150.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.155.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.157.64.0/19     *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.160.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.204.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.205.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.206.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.207.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.209.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.212.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.213.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.214.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.215.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.216.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.218.13.0/24     *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.218.14.0/24     *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.218.16.0/20     *[Static/5] 2d 12:48:34
```

```
                    > to 10.102.191.254 via fxp0.0
10.218.32.0/20      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.227.0.0/16       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
10.255.255.1/32     *[Direct/0] 2d 12:48:34
                    > via lo0.0
10.255.255.2/32     *[OSPF/10] 2d 12:48:24, metric 1
                    > to 1.0.0.2 via ge-0/0/1.0
10.255.255.4/32     *[OSPF/10] 2d 12:48:24, metric 2
                    > to 1.0.0.2 via ge-0/0/1.0
128.102.161.191/32 *[OSPF/10] 2d 12:48:24, metric 1
                    > to 1.0.0.2 via ge-0/0/1.0
128.102.169.99/32   *[Direct/0] 2d 12:48:34
                    > via lo0.0
128.102.171.41/32   *[OSPF/10] 2d 12:48:24, metric 2
                    > to 1.0.0.2 via ge-0/0/1.0
172.16.0.0/12       *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
192.168.0.0/16      *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
192.168.102.0/23    *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
207.17.136.0/24     *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
207.17.136.192/32   *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
207.17.137.0/24     *[Static/5] 2d 12:48:34
                    > to 10.102.191.254 via fxp0.0
224.0.0.5/32        *[OSPF/10] 2d 12:48:34, metric 1
                      MultiRecv

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.4/32     *[RSVP/7/1] 2d 12:48:04, metric 2
                    > to 1.0.0.2 via ge-0/0/1.0, label-switched-path to-pe2

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

47.0005.80ff.f800.0000.0108.0001.1281.0216.9099/152
                    *[Direct/0] 2d 12:48:34
```

```
                    > via lo0.0

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 2d 12:48:34, metric 1
                      Receive
1                  *[MPLS/0] 2d 12:48:34, metric 1
                      Receive
2                  *[MPLS/0] 2d 12:48:34, metric 1
                      Receive
13                 *[MPLS/0] 2d 12:48:34, metric 1
                      Receive
800003             *[L2VPN/7] 2d 12:41:29
                    > via ge-0/0/0.300, Pop       Offset: 4
ge-0/0/0.300       *[L2VPN/7] 2d 12:41:29, metric2 2
                    > to 1.0.0.2 via ge-0/0/1.0, label-switched-path to-pe2

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

abcd::128:102:169:99/128
                   *[Direct/0] 2d 12:48:34
                    > via lo0.0
fe80::5668:a60f:fc6b:eb97/128
                   *[Direct/0] 2d 12:48:34
                    > via lo0.0

bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.4:200:2:1/96
                   *[BGP/170] 2d 12:41:35, localpref 100, from 10.255.255.4
                      AS path: I, validation-state: unverified
                    > to 1.0.0.2 via ge-0/0/1.0, label-switched-path to-pe2

l2vpn-inst.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.1:200:1:1/96
                   *[L2VPN/170/-101] 2d 12:41:29, metric2 1
                      Indirect
10.255.255.4:200:2:1/96
```

```
                    *[BGP/170] 2d 12:41:35, localpref 100, from 10.255.255.4
                       AS path: I, validation-state: unverified
                     > to 1.0.0.2 via ge-0/0/1.0, label-switched-path to-pe2

l2vpn-inst.l2id.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1                   *[L2VPN/170/-101] 2d 12:41:29, metric2 1
                       Indirect
                     [L2VPN/175] 2d 12:41:29
                     > via ge-0/0/0.300, Pop       Offset: 4
2                   *[BGP/170] 2d 12:41:35, localpref 100, from 10.255.255.4
                       AS path: I, validation-state: unverified
                     > to 1.0.0.2 via ge-0/0/1.0, label-switched-path to-pe2
```

Meaning

The output shows all the routes in the routing table.

Configuring PE2

IN THIS SECTION

- Procedure | 844
- Results | 848

Procedure

Step-by-Step Procedure

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

To configure Device PE2:

1. Configure the interfaces.

```
[edit interfaces]
user@PE2# set ge-0/0/0 unit 0 family inet address 2.0.0.2/24
user@PE2# set ge-0/0/0 unit 0 family mpls
user@PE2# set ge-0/0/1 vlan-tagging
user@PE2# set ge-0/0/1 mtu 1600
user@PE2# set ge-0/0/1 encapsulation vlan-ccc
user@PE2# set ge-0/0/1 unit 300 encapsulation vlan-ccc
user@PE2# set ge-0/0/1 unit 300 vlan-id 600
user@PE2# set ge-0/0/1 unit 600 encapsulation vlan-vpls
user@PE2# set ge-0/0/1 unit 600 vlan-id 600
user@PE2# set ge-0/0/1 unit 600 family vpls deactivate interfaces ge-0/0/1 unit 600
user@PE2# set lo0 unit 0 family inet address 10.255.255.4/32
```

2. Configure the router ID.

```
[edit routing-options]
user@PE2# set router-id 10.255.255.4
```

3. Configure the autonomous system (AS) number, and apply the policy to the forwarding table of the local router with the export statement.

```
[edit routing-options]
user@PE2# set autonomous-system 100
user@PE2# set forwarding-table export exp-to-frwd
```

4. Configure the RSVP protocol on the interfaces.

```
[edit protocols rsvp]
user@PE2# set interface all
user@PE2# set interface ge-0/0/1.0
user@PE2# set interface lo0.0
```

5. Apply the label-switched path attributes to the MPLS protocol, and configure the interface.

```
[edit protocols mpls]
user@PE2# set label-switched-path to-pe1 to 10.255.255.1
user@PE2# set interface ge-0/0/0.0
```

6. Define a peer group, and configure the local-end address of the BGP session for the peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE2# set type internal
user@PE2# set local-address 10.255.255.4
```

7. Configure the attributes of the protocol family for NLRIs in updates.

```
[edit protocols bgp group vpls-pe]
user@PE2# set family l2vpn auto-discovery-only
user@PE2# set family l2vpn signaling
```

8. Configure the neighbors for peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE2# set neighbor 10.255.255.1
user@PE2# set neighbor 10.255.255.2
```

9. Configure traffic engineering, and configure the interfaces of OSPF area 0.0.0.0.

```
[edit protocols ospf]
user@PE2# set traffic-engineering
user@PE2# set area 0.0.0.0 interface lo0.0 passive
user@PE2# set area 0.0.0.0 interface ge-0/0/0.0
```

10. Configure the routing policy and the BGP community information.

```
[edit policy-options]
user@PE2# set policy-statement exp-to-frwd term 0 from community vpls-com
user@PE2# set policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe1
```

```
user@PE2# set policy-statement exp-to-frwd term 0 then accept
user@PE2# set community vpls-com members target:100:100
```

11. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances l2vpn-inst]
user@PE2# set instance-type l2vpn
user@PE2# set interface ge-0/0/1.300
```

12. Configure the route distinguisher for instance `l2vpn-inst`, and configure the VRF target community.

```
[edit routing-instances l2vpn-inst]
user@PE2# set route-distinguisher 10.255.255.4:200
user@PE2# set vrf-target target:100:100
```

13. Configure the type of encapsulation required for the L2VPN protocol.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE2# set encapsulation-type ethernet-vlan
```

14. Configure the sites connected to the provider equipment.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE2# set site pe2 site-identifier 2
user@PE2# set site pe2 interface ge-0/0/1.300 remote-site-id 1
```

15. Configure the L2VPN protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances l2vpn-inst protocols l2vpn]
user@PE2# set flow-label-transmit
user@PE2# set flow-label-receive
```

16. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances vpl1]
user@PE2# set instance-type vpls
user@PE2# set interface ge-0/0/1.600
```

17. Configure the route distinguisher for instance `vpl1`, and configure the VRF target community.

```
[edit routing-instances vpl1]
user@PE2# set route-distinguisher 10.255.255.4:100
user@PE2# set vrf-target target:100:100
```

18. Assign the maximum site identifier for the VPLS domain.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set site-range 10
```

19. Configure to not use the tunnel services for the VPLS instance, and assign a site identifier to the site connected to the provider equipment.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set no-tunnel-services
user@PE2# set site vpl1PE2 site-identifier 2
```

20. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to the push flow label in the transmit direction to the remote PE.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set flow-label-transmit
user@PE2# set flow-label-receive
```

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@PE2# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 2.0.0.2/24;
        }
        family mpls;
    }
}
ge-0/0/1 {
    vlan-tagging;
    mtu 1600;
    encapsulation vlan-ccc;
    unit 300 {
        encapsulation vlan-ccc;
        vlan-id 600;
    }
    unit 600 {
        encapsulation vlan-vpls;
        vlan-id 600;
        family vpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.4/32;
        }
    }
}
```

```
user@PE2# show protocols
rsvp {
    interface all;
    interface ge-0/0/1.0;
    interface lo0.0;
}
mpls {
```

```
    label-switched-path to-pe1 {
        to 10.255.255.1;
    }
    interface ge-0/0/0.0;
}
bgp {
    group vpls-pe {
        type internal;
        local-address 10.255.255.4;
        family l2vpn {
            auto-discovery-only;
            signaling;
        }
        neighbor 10.255.255.1;
        neighbor 10.255.255.2;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/0.0;
    }
}
```

```
user@PE2# show policy-options
policy-statement exp-to-frwd {
    term 0 {
        from community vpls-com;
        then {
            install-nexthop lsp to-pe1;
            accept;
        }
    }
```

```
}
community vpls-com members target:100:100;
```

```
user@PE2# show routing-instances
l2vpn-inst {
    instance-type l2vpn;
    interface ge-0/0/1.300;
    route-distinguisher 10.255.255.4:200;
    vrf-target target:100:100;
    protocols {
        l2vpn {
            encapsulation-type ethernet-vlan;
            site pe2 {
                site-identifier 2;
                interface ge-0/0/1.300 {
                    remote-site-id 1;
                }
            }
            flow-label-transmit;
            flow-label-receive;
        }
    }
}
 vpl1 {
    instance-type vpls;
    interface ge-0/0/1.600;
    route-distinguisher 10.255.255.4:100;
    vrf-target target:100:100;
    protocols {
        vpls {
            site-range 10;
            no-tunnel-services;
            site vpl1PE2 {
                site-identifier 2;
            }
            flow-label-transmit;
            flow-label-receive;
        }
```

```
    }
}
```

```
user@PE2# show routing-options
router-id 10.255.255.4;
autonomous-system 100;
forwarding-table {
    export exp-to-frwd;
}
```

Verification

IN THIS SECTION

- Verifying the BGP Summary Information | 852
- Verifying the L2VPN Connections Information | 853
- Verifying the Routes | 854

Confirm that the configuration is working properly.

Verifying the BGP Summary Information

Purpose

Verify the BGP summary information.

Action

From operational mode, enter the `show bgp summary` command.

```
user@PE2> show bgp summary

Groups: 1 Peers: 2 Down peers: 1
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
```

```
bgp.l2vpn.0
                       1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.255.255.1            100       8090       8119       0       1 2d 12:53:15 Establ
  bgp.l2vpn.0: 1/1/1/0
  l2vpn-inst.l2vpn.0: 1/1/1/0
10.255.255.2            100          0          0       0       0 2d 14:14:49 Active
```

Meaning

The output displays the BGP summary information.

Verifying the L2VPN Connections Information

Purpose

Verify the Layer 2 VPN connections information.

Action

From operational mode, run the `show l2vpn connections` command to display the Layer 2 VPN connections information.

```
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        SN -- Static Neighbor
LB -- Local site not best-site   RB -- Remote site not best-site
VM -- VLAN ID mismatch

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

Instance: l2vpn-inst
Edge protection: Not-Primary
  Local site: pe2 (2)
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Jun 22 14:46:50 2015           1
      Remote PE: 10.255.255.1, Negotiated control-word: Yes (Null)
      Incoming label: 800002, Outgoing label: 800003
      Local interface: ge-0/0/1.300, Status: Up, Encapsulation: VLAN
      Flow Label Transmit: Yes, Flow Label Receive: Yes
```

Meaning

The output displays the Layer 2 VPN connections information along with the flow label transmit and flow label receive information.

Verifying the Routes

Purpose

Verify that the expected routes are learned.

Action

From operational mode, run the `show route` command to display the routes in the routing table.

```
user@PE2> show route

inet.0: 51 destinations, 51 routes (51 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
1.0.0.0/24          *[OSPF/10] 2d 14:09:33, metric 2
                     > to 2.0.0.1 via ge-0/0/0.0
2.0.0.0/24          *[Direct/0] 2d 14:10:18
                     > via ge-0/0/0.0
2.0.0.2/32          *[Local/0] 2d 14:10:20
                       Local via ge-0/0/0.0
10.4.0.0/16         *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.5.0.0/16         *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.6.128.0/17       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.9.0.0/16         *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.10.0.0/16        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.13.4.0/23        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.13.10.0/23       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.82.0.0/15        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.84.0.0/16        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.85.12.0/22       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.92.0.0/16        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.94.0.0/16        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.99.0.0/16        *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.102.0.0/16       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.102.160.0/19     *[Direct/0] 2d 14:12:18
                     > via fxp0.0
10.102.171.41/32    *[Local/0] 2d 14:12:18
                       Local via fxp0.0
10.150.0.0/16       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.155.0.0/16       *[Static/5] 2d 14:12:18
                     > to 10.102.191.254 via fxp0.0
10.157.64.0/19      *[Static/5] 2d 14:12:18
```

```
                    > to 10.102.191.254 via fxp0.0
10.160.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.204.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.205.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.206.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.207.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.209.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.212.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.213.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.214.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.215.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.216.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.218.13.0/24      *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.218.14.0/24      *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.218.16.0/20      *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.218.32.0/20      *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.227.0.0/16       *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
10.255.255.1/32     *[OSPF/10] 2d 12:50:36, metric 2
                    > to 2.0.0.1 via ge-0/0/0.0
10.255.255.2/32     *[OSPF/10] 2d 14:09:33, metric 1
                    > to 2.0.0.1 via ge-0/0/0.0
10.255.255.4/32     *[Direct/0] 2d 14:11:51
                    > via lo0.0
128.102.161.191/32 *[OSPF/10] 2d 14:09:33, metric 1
                    > to 2.0.0.1 via ge-0/0/0.0
128.102.169.99/32   *[OSPF/10] 2d 12:50:36, metric 2
                    > to 2.0.0.1 via ge-0/0/0.0
```

```
128.102.171.41/32  *[Direct/0] 2d 14:12:18
                    > via lo0.0
172.16.0.0/12      *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
192.168.0.0/16     *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
192.168.102.0/23   *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
207.17.136.0/24    *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
207.17.136.192/32  *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
207.17.137.0/24    *[Static/5] 2d 14:12:18
                    > to 10.102.191.254 via fxp0.0
224.0.0.5/32       *[OSPF/10] 2d 14:11:51, metric 1
                      MultiRecv

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.1/32    *[RSVP/7/1] 2d 12:50:24, metric 2
                    > to 2.0.0.1 via ge-0/0/0.0, label-switched-path to-pe1

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

47.0005.80ff.f800.0000.0108.0001.1281.0217.1041/152
                   *[Direct/0] 2d 14:12:18
                    > via lo0.0

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 2d 14:11:51, metric 1
                      Receive
1                  *[MPLS/0] 2d 14:11:51, metric 1
                      Receive
2                  *[MPLS/0] 2d 14:11:51, metric 1
                      Receive
13                 *[MPLS/0] 2d 14:11:51, metric 1
                      Receive
800002             *[L2VPN/7] 2d 12:43:43
                    > via ge-0/0/1.300, Pop       Offset: 4
```

```
ge-0/0/1.300       *[L2VPN/7] 2d 12:43:43, metric2 2
                    > to 2.0.0.1 via ge-0/0/0.0, label-switched-path to-pe1

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

abcd::128:102:171:41/128
                   *[Direct/0] 2d 14:12:18
                    > via lo0.0
fe80::5668:a60f:fc6b:ee28/128
                   *[Direct/0] 2d 14:12:18
                    > via lo0.0

bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.1:200:1:1/96
                   *[BGP/170] 2d 12:43:43, localpref 100, from 10.255.255.1
                      AS path: I, validation-state: unverified
                    > to 2.0.0.1 via ge-0/0/0.0, label-switched-path to-pe1

l2vpn-inst.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.255.1:200:1:1/96
                   *[BGP/170] 2d 12:43:43, localpref 100, from 10.255.255.1
                      AS path: I, validation-state: unverified
                    > to 2.0.0.1 via ge-0/0/0.0, label-switched-path to-pe1
10.255.255.4:200:2:1/96
                   *[L2VPN/170/-101] 2d 12:43:50, metric2 1
                      Indirect

l2vpn-inst.l2id.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1                  *[BGP/170] 2d 12:43:43, localpref 100, from 10.255.255.1
                      AS path: I, validation-state: unverified
                    > to 2.0.0.1 via ge-0/0/0.0, label-switched-path to-pe1
2                  *[L2VPN/170/-101] 2d 12:43:50, metric2 1
                      Indirect
                    [L2VPN/175] 2d 12:43:43
```

```
                    > via ge-0/0/1.300, Pop       Offset: 4
```

Meaning

The output shows all the routes in the routing table.

SEE ALSO

- Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **823**
- Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **860**
- FAT Pseudowire Support for BGP L2VPN and VPLS Overview | **822**
- *flow-label-receive*
- *flow-label-transmit*

Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic

The flow-aware transport (FAT) or flow label is supported for BGP-signaled pseudowires such as VPLS and is to be configured only on the label edge routers (LERs). This enables the transit routers or the label-switching routers (LSRs) to perform load balancing of MPLS packets across equal-cost multipath (ECMP) or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. FAT pseudowires or flow label can be used with LDP-signaled VPLS with forwarding equivalence class (FEC128 and FEC129), and the support for flow label is extended for BGP-signaled pseudowires for point-to-point or point-to-multipoint Layer 2 services.

Before you configure FAT pseudowire support for BGP VPLS to load-balance MPLS traffic:

- Configure the device interfaces and enable MPLS on all the interfaces.
- Configure RSVP.
- Configure MPLS and an LSP to the remote PE router.
- Configure BGP and OSPF.

To configure FAT pseudowire support for BGP VPLS to load-balance MPLS traffic, you must do the following:

1. Configure the sites connected to the provider equipment for a given routing instance for the VPLS protocols.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set site site-name site-identifier site-identifier
user@host#  set site-range site-range
```

2. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set flow-label-receive
```

3. Configure the VPLS protocol to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances routing-instance name protocols vpls]
user@host#  set flow-label-transmit
```

SEE ALSO

FAT Pseudowire Support for BGP L2VPN and VPLS Overview | 822

Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | 823

Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | 824

flow-label-receive

flow-label-transmit

Example: Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic

IN THIS SECTION

Requirements | 861

- Overview | **861**
- Configuration | **862**
- Verification | **879**

This example shows how to implement FAT pseudowire support for BGP VPLS to help load-balance MPLS traffic.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers
- Junos OS Release 16.1 or later running on all devices

Before you configure FAT pseudowire support for BGP VPLS, be sure you configure the routing and signaling protocols.

Overview

IN THIS SECTION

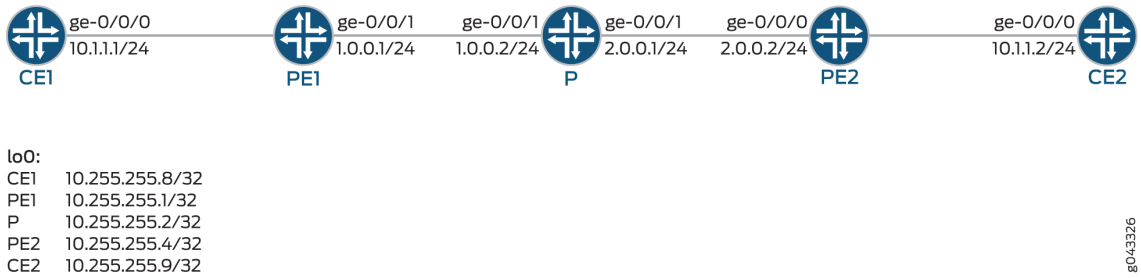
- Topology | **861**

Junos OS allows the flow-aware transport (FAT) flow label that is supported for BGP-signaled pseudowires such as VPLS to be configured only on the label edge routers (LERs). This causes the transit routers or the label-switching routers (LSRs) to perform load balancing of MPLS packets across equal-cost multipath (ECMP) paths or link aggregation groups (LAGs) without the need for deep packet inspection of the payload. The FAT flow label can be used for LDP-signaled forwarding equivalence class (FEC 128 and FEC 129) pseudowires for VPWS and VPLS pseudowires.

Topology

Figure 57 on page 862 shows the FAT pseudowire support for BGP VPLS configured on Device PE1 and Device PE2.

Figure 57: Example FAT Pseudowire Support for BGP VPLS



Configuration

IN THIS SECTION

- Verification | 877

CLI Quick Configuration

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

CE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family inet address 10.1.1.1/24
set interfaces lo0 unit 0 family inet address 10.255.255.8/32
```

PE1

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mtu 1600
set interfaces ge-0/0/0 encapsulation vlan-vpls
set interfaces ge-0/0/0 unit 600 encapsulation vlan-vpls
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family vpls
```

```
set interfaces ge-0/0/1 unit 0 family inet address 1.0.0.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.1/32
set routing-options nonstop-routing
set routing-options router-id 10.255.255.1
set routing-options autonomous-system 100
set routing-options forwarding-table export exp-to-frwd
set protocols rsvp interface all
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols mpls label-switched-path to-pe2 to 10.255.255.4
set protocols mpls interface ge-0/0/1.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.1
set protocols bgp group vpls-pe family l2vpn auto-discovery-only
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.4
set protocols bgp group vpls-pe neighbor 10.255.255.2
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
set policy-options policy-statement exp-to-frwd term 0 from community vpls-com
set policy-options policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe2
set policy-options policy-statement exp-to-frwd term 0 then accept
set policy-options community vpls-com members target:100:100
set routing-instances vpl1 instance-type vpls
set routing-instances vpl1 interface ge-0/0/0.600
set routing-instances vpl1 route-distinguisher 10.255.255.1:100
set routing-instances vpl1 vrf-target target:100:100
set routing-instances vpl1 protocols vpls site-range 10
set routing-instances vpl1 protocols vpls no-tunnel-services
set routing-instances vpl1 protocols vpls site vpl1PE1 site-identifier 1
set routing-instances vpl1 protocols vpls flow-label-transmit
set routing-instances vpl1 protocols vpls flow-label-receive
```

P

```
set interfaces ge-0/0/0 unit 0 family inet address 1.0.0.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 2.0.0.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.255.2/32
```

```
set routing-options router-id 10.255.255.2
set routing-options autonomous-system 100
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface lo0.0
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.2
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.1
set protocols bgp group vpls-pe neighbor 10.255.255.4 deactivate protocols bgp
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
```

PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 2.0.0.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 mtu 1600
set interfaces ge-0/0/1 encapsulation vlan-vpls
set interfaces ge-0/0/1 unit 600 encapsulation vlan-vpls
set interfaces ge-0/0/1 unit 600 vlan-id 600
set interfaces ge-0/0/1 unit 600 family vpls
set interfaces lo0 unit 0 family inet address 10.255.255.4/32
set routing-options router-id 10.255.255.4
set routing-options autonomous-system 100
set routing-options forwarding-table export exp-to-frwd
set protocols rsvp interface all
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
set protocols mpls label-switched-path to-pe1 to 10.255.255.1
set protocols mpls interface ge-0/0/0.0
set protocols bgp group vpls-pe type internal
set protocols bgp group vpls-pe local-address 10.255.255.4
set protocols bgp group vpls-pe family l2vpn auto-discovery-only
set protocols bgp group vpls-pe family l2vpn signaling
set protocols bgp group vpls-pe neighbor 10.255.255.1
set protocols bgp group vpls-pe neighbor 10.255.255.2
```

```
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set policy-options policy-statement exp-to-frwd term 0 from community vpls-com
set policy-options policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe1
set policy-options policy-statement exp-to-frwd term 0 then accept
set policy-options community vpls-com members target:100:100
set routing-instances vpl1 instance-type vpls
set routing-instances vpl1 interface ge-0/0/1.600
set routing-instances vpl1 route-distinguisher 10.255.255.4:100
set routing-instances vpl1 vrf-target target:100:100
set routing-instances vpl1 protocols vpls site-range 10
set routing-instances vpl1 protocols vpls no-tunnel-services
set routing-instances vpl1 protocols vpls site vpl1PE2 site-identifier 2
set routing-instances vpl1 protocols vpls flow-label-transmit
set routing-instances vpl1 protocols vpls flow-label-receive
```

CE2

```
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 unit 600 vlan-id 600
set interfaces ge-0/0/0 unit 600 family inet address 10.1.1.2/24
set interfaces lo0 unit 0 family inet address 10.255.255.9/32
```

Configuring PE1

Step-by-Step Procedure

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

To configure Device PE1:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-0/0/0 vlan-tagging
user@PE1# set ge-0/0/0 mtu 1600
user@PE1# set ge-0/0/0 encapsulation vlan-vpls
user@PE1# set ge-0/0/0 unit 600 encapsulation vlan-vpls
```

```
user@PE1# set ge-0/0/0 unit 600 vlan-id 600
user@PE1# set ge-0/0/0 unit 600 family vpls
user@PE1# set ge-0/0/1 unit 0 family inet address 1.0.0.1/24
user@PE1# set ge-0/0/1 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.255.255.1/32
```

2. Configure nonstop routing, and configure the router ID.

```
[edit routing-options]
user@PE1# set nonstop-routing
user@PE1# set router-id 10.255.255.1
```

3. Configure the autonomous system (AS) number, and apply the policy to the forwarding table of the local router with the export statement.

```
[edit routing-options]
user@PE1# set autonomous-system 100
user@PE1# set forwarding-table export exp-to-frwd
```

4. Configure the RSVP protocol on the interfaces.

```
[edit protocols rsvp]
user@PE1# set interface all
user@PE1# set interface ge-0/0/1.0
user@PE1# set interface lo0.0
```

5. Apply the label-switched path attributes to the MPLS protocol, and configure the interface.

```
[edit protocols mpls]
user@PE1# set label-switched-path to-pe2 to 10.255.255.4
user@PE1# set interface ge-0/0/1.0
```

6. Define a peer group, and configure the address of the local end of the BGP session for peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE1# set type internal
user@PE1# set local-address 10.255.255.1
```

7. Configure attributes of the protocol family for NLRIs in updates.

```
[edit protocols bgp group vpls-pe family l2vpn]
user@PE1# set auto-discovery-only
user@PE1# set signaling
```

8. Configure neighbors for the peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE1# set neighbor 10.255.255.4
user@PE1# set neighbor 10.255.255.2
```

9. Configure traffic engineering, and configure the interfaces of OSPF area 0.0.0.0.

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

10. Configure the routing policy and the BGP community information.

```
[edit policy-options ]
user@PE1# set policy-statement exp-to-frwd term 0 from community vpls-com
user@PE1# set policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe2
user@PE1# set policy-statement exp-to-frwd term 0 then accept
user@PE1# set community vpls-com members target:100:100
```

11. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances vp1]
user@PE1# set instance-type vpls
user@PE1# set interface ge-0/0/0.600
```

12. Configure the route distinguisher for instance `vpl1`, and configure the VRF target community.

```
[edit routing-instances vpl1]
user@PE1# set route-distinguisher 10.255.255.1:100
user@PE1# set vrf-target target:100:100
```

13. Assign the maximum site identifier for the VPLS domain.

```
[edit routing-instances vp1 protocols vpls]
user@PE1# set site-range 10
```

14. Configure the VPLS protocol to not use the tunnel services for the VPLS instance, and assign the site identifier to the site connected to the provider equipment.

```
[edit routing-instances vp1 protocols vpls]
user@PE1# set no-tunnel-services
user@PE1# set site vpl1PE1 site-identifier 1
```

15. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances vp1 protocols vpls]
user@PE1# set flow-label-receive
user@PE1# set flow-label-transmit
```

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-0/0/0 {
    vlan-tagging;
    mtu 1600;
    encapsulation vlan-vpls;
    unit 600 {
        encapsulation vlan-vpls;
        vlan-id 600;
        family vpls;
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 1.0.0.1/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.1/32;
        }
    }
}
```

```
user@PE1# show protocols
rsvp {
    interface all;
    interface ge-0/0/1.0;
    interface lo0.0;
}
mpls {
    label-switched-path to-pe2 {
```

```
        to 10.255.255.4;
    }
    interface ge-0/0/1.0;
}
bgp {
    group vpls-pe {
        type internal;
        local-address 10.255.255.1;
        family l2vpn {
            auto-discovery-only;
            signaling;
        }
        neighbor 10.255.255.4;
        neighbor 10.255.255.2;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/1.0;
    }
}
```

```
user@PE1# show policy-options
policy-statement exp-to-frwd {
    term 0 {
        from community vpls-com;
        then {
            install-nexthop lsp to-pe2;
            accept;
        }
    }
}
community vpls-com members target:100:100;
```

```
user@PE1# show routing-instances
vpl1 {
```

```
    instance-type vpls;
    interface ge-0/0/0.600;
    route-distinguisher 10.255.255.1:100;
    vrf-target target:100:100;
    protocols {
        vpls {
            site-range 10;
            no-tunnel-services;
            site vpl1PE1 {
                site-identifier 1;
            }
            flow-label-transmit;
            flow-label-receive;
        }
    }
}
```

```
user@PE1# show routing-options
nonstop-routing;
router-id 10.255.255.1;
autonomous-system 100;
forwarding-table {
    export exp-to-frwd;
}
```

Configuring PE2

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 Junos OS CLI User Guide.

To configure Device PE2:

1. Configure the interfaces.

```
[edit interfaces]
user@PE2# set ge-0/0/0 unit 0 family inet address 2.0.0.2/24
user@PE2# set ge-0/0/0 unit 0 family mpls
user@PE2# set ge-0/0/1 vlan-tagging
```

```
user@PE2# set ge-0/0/1 mtu 1600
user@PE2# set ge-0/0/1 encapsulation vlan-vpls
user@PE2# set ge-0/0/1 unit 600 encapsulation vlan-vpls
user@PE2# set ge-0/0/1 unit 600 vlan-id 600
user@PE2# set ge-0/0/1 unit 600 family vpls
user@PE2# set lo0 unit 0 family inet address 10.255.255.4/32
```

2. Configure the router ID.

```
[edit routing-options]
user@PE2# set router-id 10.255.255.4
```

3. Configure the autonomous system (AS) number, and apply the policy to the forwarding table of the local router with the export statement.

```
[edit routing-options]
user@PE2# set autonomous-system 100
user@PE2# set forwarding-table export exp-to-frwd
```

4. Configure the RSVP protocol on the interfaces.

```
[edit protocols rsvp]
user@PE2# set interface all
user@PE2# set interface ge-0/0/1.0
user@PE2# set interface lo0.0
```

5. Apply the label-switched path attributes to the MPLS protocol, and configure the interface.

```
[edit protocols mpls]
user@PE2# set label-switched-path to-pe1 to 10.255.255.1
user@PE2# set interface ge-0/0/0.0
```

6. Define a peer group, and configure the local-end address of the BGP session for peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE2# set type internal
user@PE2# set local-address 10.255.255.4
```

7. Configure attributes of the protocol family for NLRIs in updates.

```
[edit protocols bgp group vpls-pe]
user@PE2# set family l2vpn auto-discovery-only
user@PE2# set family l2vpn signaling
```

8. Configure neighbors for the peer group `vpls-pe`.

```
[edit protocols bgp group vpls-pe]
user@PE2# set neighbor 10.255.255.1
user@PE2# set neighbor 10.255.255.2
```

9. Configure traffic engineering, and configure the interfaces of OSPF area 0.0.0.0.

```
[edit protocols ospf]
user@PE2# set traffic-engineering
user@PE2# set area 0.0.0.0 interface lo0.0 passive
user@PE2# set area 0.0.0.0 interface ge-0/0/0.0
```

10. Configure the routing policy and the BGP community information.

```
[edit policy-options ]
user@PE2# set policy-statement exp-to-frwd term 0 from community vpls-com
user@PE2# set policy-statement exp-to-frwd term 0 then install-nexthop lsp to-pe1
user@PE2# set policy-statement exp-to-frwd term 0 then accept
user@PE2# set community vpls-com members target:100:100
```

11. Configure the type of routing instance, and configure the interface.

```
[edit routing-instances vpl1]
user@PE2# set instance-type vpls
user@PE2# set interface ge-0/0/1.600
```

12. Configure the route distinguisher for instance vp11, and configure the VRF target community.

```
[edit routing-instances vpl1]
user@PE2# set route-distinguisher 10.255.255.4:100
user@PE2# set vrf-target target:100:100
```

13. Assign the maximum site identifier for the VPLS domain.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set site-range 10
```

14. Configure the VPLS protocol to not use the tunnel services for the VPLS instance, and assign the site identifier to the site connected to the provider equipment.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set no-tunnel-services
user@PE2# set site vpl1PE2 site-identifier 2
```

15. Configure the VPLS protocol for the routing instance to provide advertising capability to pop the flow label in the receive direction to the remote PE and to provide advertising capability to push the flow label in the transmit direction to the remote PE.

```
[edit routing-instances vpl1 protocols vpls]
user@PE2# set flow-label-transmit
user@PE2# set flow-label-receive
```

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@PE2# show interfaces
ge-0/0/0 {
```

```
    unit 0 {
        family inet {
            address 2.0.0.2/24;
        }
        family mpls;
    }
}
ge-0/0/1 {
    vlan-tagging;
    mtu 1600;
    encapsulation vlan-vpls;
    unit 600 {
        encapsulation vlan-vpls;
        vlan-id 600;
        family vpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.255.4/32;
        }
    }
}
```

```
user@PE2# show protocols
rsvp {
    interface all;
    interface ge-0/0/1.0;
    interface lo0.0;
}
mpls {
    label-switched-path to-pe1 {
        to 10.255.255.1;
    }
    interface ge-0/0/0.0;
}
bgp {
    group vpls-pe {
        type internal;
        local-address 10.255.255.4;
```

```
        family l2vpn {
            auto-discovery-only;
            signaling;
        }
        neighbor 10.255.255.1;
        neighbor 10.255.255.2;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/0.0;
    }
}
```

```
user@PE2# show policy-options
policy-statement exp-to-frwd {
    term 0 {
        from community vpls-com;
        then {
            install-nexthop lsp to-pe1;
            accept;
        }
    }
}
community vpls-com members target:100:100;
```

```
user@PE2# show routing-instances
vpl1 {
    instance-type vpls;
    interface ge-0/0/1.600;
    route-distinguisher 10.255.255.4:100;
    vrf-target target:100:100;
    protocols {
        vpls {
            site-range 10;
            no-tunnel-services;
```

```
            site vpl1PE2 {
                site-identifier 2;
            }
            flow-label-transmit;
            flow-label-receive;
        }
    }
}
```

```
user@PE2# show routing-options
router-id 10.255.255.4;
autonomous-system 100;
forwarding-table {
    export exp-to-frwd;
}
```

Verification

IN THIS SECTION

- Verifying the VPLS Connection Information | **877**

Confirm that the configuration is working properly.

Verifying the VPLS Connection Information

Purpose

Verify the VPLS connection information.

Action

From operational mode, run the `show vpls connections` command to display the VPLS connections information.

```
user@PE1> show vpls 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        SN -- Static Neighbor
LB -- Local site not best-site   RB -- Remote site not best-site
VM -- VLAN ID mismatch

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

Instance: vpl1
Edge protection: Not-Primary
  Local site: vpl1PE1 (1)
    connection-site           Type  St     Time last up          # Up trans
    2                         rmt   Up     Jun 17 11:38:14 2015           1
      Remote PE: 10.255.255.4, Negotiated control-word: No
      Incoming label: 262146, Outgoing label: 262145
      Local interface: lsi.1048576, Status: Up, Encapsulation: VPLS
```

```
    Description: Intf - vpls vpl1 local site 1 remote site 2
  Flow Label Transmit: Yes, Flow Label Receive: Yes
```

Meaning

The output displays the VPLS connection information along with the flow label receive and flow label transmit information.

Verification

IN THIS SECTION

- Verifying the VPLS Connection Information | **879**

Confirm that the configuration is working properly.

Verifying the VPLS Connection Information

Purpose

Verify the VPLS connection information.

Action

From operational mode, run the `show vpls connections` command to display the VPLS connections information.

```
user@PE2> show vpls 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        SN -- Static Neighbor
LB -- Local site not best-site   RB -- Remote site not best-site
VM -- VLAN ID mismatch

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

Instance: vpl1
Edge protection: Not-Primary
  Local site: vpl1PE2 (2)
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Jun 17 11:38:14 2015           1
      Remote PE: 10.255.255.1, Negotiated control-word: No
      Incoming label: 262145, Outgoing label: 262146
      Local interface: lsi.1048576, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls vpl1 local site 2 remote site 1
      Flow Label Transmit: Yes, Flow Label Receive: Yes
```

Meaning

The output displays the VPLS connection information along with the flow label receive and flow label transmit information.

SEE ALSO

- Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **823**
- Configuring FAT Pseudowire Support for BGP VPLS to Load-Balance MPLS Traffic | **859**
- Example: Configuring FAT Pseudowire Support for BGP L2VPN to Load-Balance MPLS Traffic | **824**
- *flow-label-receive*
- *flow-label-transmit*

Change History Table

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

Release	Description
20.2R1	Starting from Junos OS Release 20.2R1, MX Series, EX9204, EX9208, EX9214, EX9251, and EX9253 devices support BGP PIC Edge protection for Layer 2 circuit, Layer 2 VPN, and VPLS (BGP VPLS, LDP VPLS and FEC 129 VPLS) services with BGP labeled unicast as the transport protocol.
19.2R1	Starting with Junos OS Release 19.2R1, you can specify a maximum number of 512 equal-cost paths on QFX10000 switches.
19.1R1	Starting with Junos OS Release 19.1R1, you can specify a maximum number of 128 equal-cost paths on QFX10000 switches.
18.4R1	Starting in Junos OS Release 18.4R1, BGP can advertise a maximum of 2 add-path routes in addition to the multiple ECMP paths.
18.1R1	Starting in Junos OS Release 18.1R1 BGP multipath is supported globally at `[edit protocols bgp]` hierarchy level. You can selectively disable multipath on some BGP groups and neighbors. Include `disable` at `[edit protocols bgp group` *`group-name`* `multipath]` hierarchy level to disable multipath option for a group or a specific BGP neighbor.
18.1R1	Starting in Junos OS Release 18.1R1, you can defer multipath calculation until all BGP routes are received. When multipath is enabled, BGP inserts the route into the multipath queue each time a new route is added or whenever an existing route changes. When multiple paths are received through BGP add-path feature, BGP might calculate one multipath route multiple times. Multipath calculation slows down the RIB (also known as the routing table) learning rate. To speed up RIB learning, multipath calculation can be either deferred until the BGP routes are received or you can lower the priority of the multipath build job as per your requirements until the BGP routes are resolved. To defer the multipath calculation configure `defer-initial-multipath-build` at `[edit protocols bgp]` hierarchy level. Alternatively, you can lower the BGP multipath build job priority using `multipath-build-priority` configuration statement at `[edit protocols bgp]` hierarchy level to speed up RIB learning.

BGP Egress Traffic Engineering

IN THIS SECTION

- Egress Peer Traffic Engineering Using BGP Labeled Unicast Overview | **882**
- Configuring Egress Peer Traffic Engineering by Using BGP Labeled Unicast and Enabling MPLS Fast Reroute | **883**
- Example: Configuring Egress Peer Traffic Engineering Using BGP Labeled Unicast | **886**
- Segment Routing Traffic Engineering at BGP Ingress Peer Overview | **911**
- Configuring Ingress Traffic Engineering with Segment Routing in a BGP Network | **915**
- Enabling Traffic Statistics Collection for BGP Labeled Unicast | **919**
- Understanding SRv6 Network Programming and Layer 3 Services over SRv6 in BGP | **921**
- Example: Configuring Layer 3 Services over SRv6 in BGP Networks | **924**
- Understanding SR-TE Policy for SRv6 Tunnel | **948**
- Example: Configuring Static SR-TE Policy for an SRv6 Tunnel | **953**

Egress Peer Traffic Engineering Using BGP Labeled Unicast Overview

In a data center environment, which mimics an ISP BGP-free core, the ingress nodes tunnel the service traffic to an egress router that is also the AS boundary router. Egress peer traffic engineering allows a central controller to instruct an ingress router in a domain to direct the traffic towards a specific egress router and a specific external interface to reach a particular destination out of the network. Egress peer traffic engineering allows for the selection of the best advertised egress route and mapping of the selected best route to a specific egress point. In case of load balancing at the ingress, this feature ensures optimum utilization of the advertised egress routes.

The ingress router controls the egress peer selection by pushing the corresponding MPLS label on an MPLS label stack for traffic engineering the links between ASs. AS boundary routers automatically install the IPv4 or IPv6 peer /32 or /128 route to an established external BGP peer that is configured with the egress traffic engineering feature into the `inet.3` forwarding table. These routes have a forwarding action of pop and forward, that is, remove the label, and forward the packet to the external BGP peer.

AS boundary routers advertise the IPv4 or IPv6 peer /32 or/128 route to the ingress BGP peers with self IPv4 next hop. Ingress BGP peers have a transport tunnel, such as MPLS LDP to reach the AS boundary router. Thus, all the network exit points are advertised to the MPLS network cloud as labeled

BGP routes. The AS boundary routers advertise service routes with these exit points as protocol next hops. The AS boundary routers readvertise the service routes from the external BGP peers towards the core without altering the next-hop addresses. However, the ingress routers resolve the protocol next hop in the service routes to map to the correct transport tunnel to the egress peer interface. Thus, the ingress routers map traffic for a specific service prefix to a specific egress router or load-balance the traffic across available egress devices. This feature allows the ingress router to direct the service traffic towards a specific egress peer.

In addition to egress peer traffic engineering, this feature provides MPLS fast reroute (FRR) for each egress device it advertises to the MPLS IPv4 network cloud. You can configure one or more backup devices for the primary egress AS boundary router. Junos OS automatically installs the backup path in addition to the primary path into the MPLS forwarding table of the established egress BGP peer that has egress peer traffic engineering configured. The AS boundary router switches to the backup path when the primary link fails and provides MPLS FRR . The specified backup path is through another directly connected external BGP peer or a remote next hop. You can also configure a backup path using ip lookup in an `inet6.0` table. However, the `remote-nexthop` and `ip-forward` backup options are mutually exclusive.

SEE ALSO

> Configuring Egress Peer Traffic Engineering by Using BGP Labeled Unicast and Enabling MPLS Fast Reroute | **883**
>
> *egress-te*
>
> *egress-te-backup-paths*

Configuring Egress Peer Traffic Engineering by Using BGP Labeled Unicast and Enabling MPLS Fast Reroute

Egress peer traffic engineering (TE) allows a central controller to instruct an ingress router in a domain to direct traffic towards a specific egress router and a specific external interface to reach a particular destination out of the network for optimum utilization of the advertised egress routes during load balancing.

BGP segregates the network into layers, such as transport and service layers. The BGP labeled unicasts form the transport layer, and the BGP unicast subsequent address family identifier (SAFI) add path routes form the service layer. The AS boundary router triggers the transport layer BGP labeled unicast label-switched paths (LSPs) that provide a route to the egress peers. The service layer add path routes use these egress peers as protocol next hop. The AS boundary routers optionally provide MPLS fast reroute (FRR) at the transport layer, which must be utilized because service layer peering issues are common. Therefore, you can specify one or more backup devices for the primary egress AS boundary

router. Junos OS automatically installs the backup path in addition to the primary path into the MPLS forwarding table of the established egress BGP peer that has egress peer TE configured. The backup path provides FRR when the primary link fails.

1. To enable egress peer TE using BGP labeled unicast:

   Enable egress peer TE at the AS boundary router for the egress BGP peer.

   ```
   [edit protocols bgp group group-name neighbor address]
   user@host# set egress-te
   ```

   For example, enable egress peer TE on the egress BGP peer.

   ```
   [edit protocols bgp group Peer1-lan-1 neighbor 200.200.201.1]
   user@host# set egress-te
   ```

2. To enable FRR for the egress traffic on BGP labeled unicast LSP:

   a. Define a template with backup paths on the egress BGP peer to enable MPLS fast reroute.

      You can define more than one template and several BGP groups, or peers can use the same defined template. All addresses listed in one template must belong to the same IP address family as the egress BGP peer.

      ```
      [edit protocols bgp ]
      user@host# set egress-te-backup-paths template backup-path
      ```

      For example, define a backup path template to enable MPLS fast reroute.

      ```
      [edit protocols bgp ]
      user@host# set egress-te-backup-paths template Customer1
      ```

   b. Configure another directly connected external BGP peer as a backup path.

      ```
      [edit protocols bgp egress-te-backup-paths template backup-path]
      user@host# set peer peer-addr
      ```

For example, configure the peer backup path for the defined template *customer1*.

```
[edit protocols bgp egress-te-backup-paths template customer1]
user@host# set peer 200.200.0.1
```

c. Configure IP forwarding on the AS boundary router as the fast reroute backup path.

Junos OS looks up the backup path in the `inet6.0` table.

You can specify the routing instance for which you are configuring backup paths on the egress BGP peer. If you do not specify a routing instance, the device configures the backup path for the master instance. Optionally, you can configure a foo routing instance as the `ip-forward` backup option.

You cannot use this option with the `remote-nexthop` option.

```
[edit protocols bgp egress-te-backup-paths template backup-path]
user@host# set ip-forward rti-name
```

For example, configure ip forwarding instance foo for the defined template *customer1*.

```
[edit protocols bgp egress-te-backup-paths template customer1]
user@host# set ip-forward foo
```

Junos OS looks up the backup path in the `foo.inet6.0` table.

d. Specify a remote next-hop address as the backup path for the egress BGP peer.

The egress peer TE AS boundary router tunnels the traffic to this remote next-hop address.

```
[edit protocols bgp egress-te-backup-paths template backup-path]
user@host# set remote-nexthop remote-nh-addr
```

For example, if you want to configure a remote next hop for the defined template *customer1*, enter:

```
[edit protocols bgp egress-te-backup-paths template customer1]
user@host# set remote-nexthop 100.100.0.1
```

e. Specify the defined template at a BGP group or neighbor level.

```
[edit protocols bgp group group-name neighbor address]
user@host# set egress-te
user@host# set backup-path backup-path
```

For example, specify the template *customer1* defined previously as the backup-path for BGP neighbor 200.200.201.1.

```
[edit protocols bgp group Peer1-lan-1 neighbor 200.200.201.1]
user@host# set egress-te
user@host# set backup-path customer1
```

SEE ALSO

Example: Configuring Egress Peer Traffic Engineering Using BGP Labeled Unicast | **886**

egress-te

egress-te-backup-paths

Example: Configuring Egress Peer Traffic Engineering Using BGP Labeled Unicast

IN THIS SECTION

- Requirements | **887**
- Overview | **887**
- Configuration | **888**
- Verification | **907**

This example shows how to configure egress peer traffic engineering using BGP labeled unicast. Egress peer traffic engineering allows a central controller to instruct an ingress router in a domain to direct traffic towards a specific egress router and a specific external interface to reach a particular destination

out of the network. In case of load balancing at the ingress, this feature ensures optimum utilization of the advertised egress routes.

Requirements

This example uses the following hardware and software components:

- Nine MX Series routers
- Junos OS Release 14.2R4 or later

Overview

IN THIS SECTION

- Topology | 887

Beginning with Junos OS Release 14.2R4, you can enable traffic engineering (TE) of service traffic, such as MPLS LSP traffic between autonomous systems (ASs) using BGP labeled unicast for optimum utilization of the advertised egress routes during load balancing.

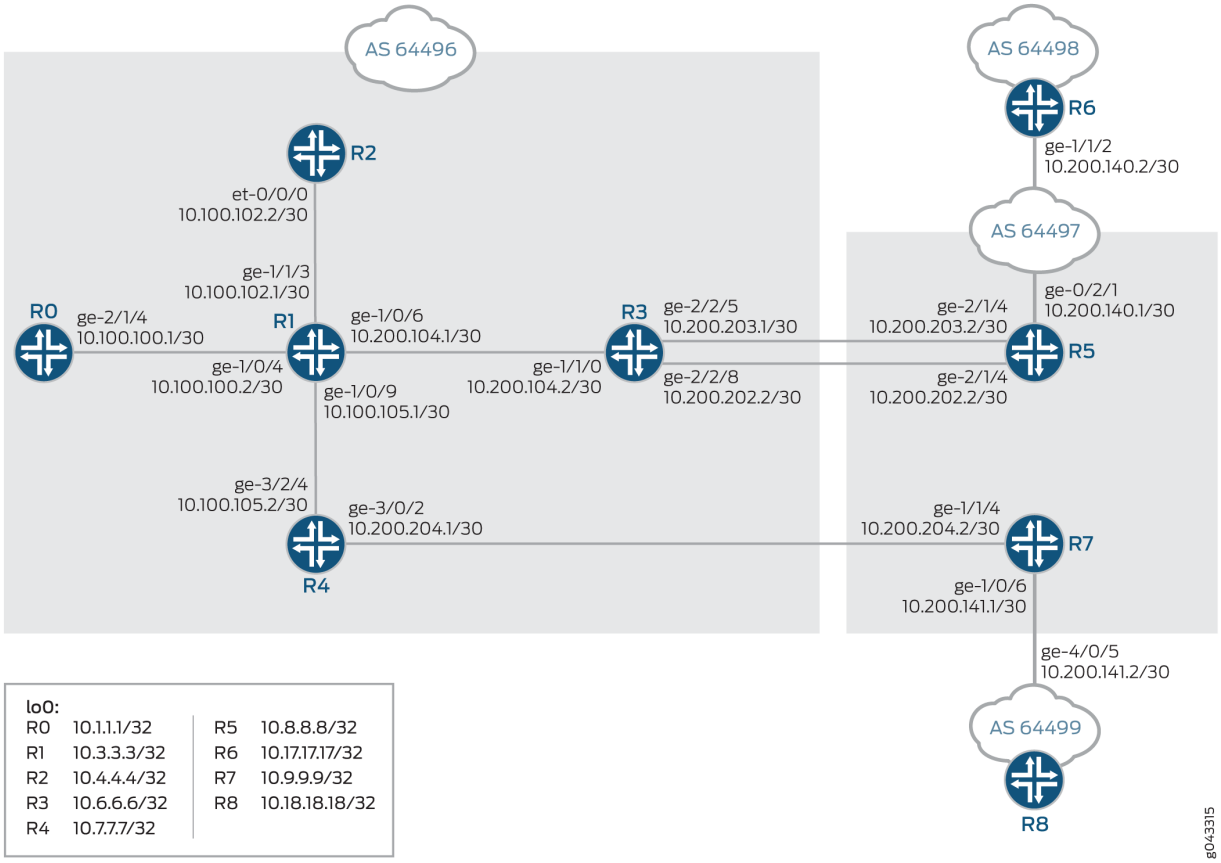
Configure egress peer TE to direct core service traffic such as MPLS RSVP to a specific egress BGP peer. The ingress BGP peer can traffic-engineer the core inet unicast and inet6 unicast service traffic using BGP labeled unicast towards a specific egress BGP peer.

NOTE: You cannot configure egress peer TE for external BGP multihop peers. The ARP routes in `inet.3` are installed for peer /32 and /128 routes only.

Topology

Figure 58 on page 888 shows the sample topology. Router R3 and Router R4 are the AS boundary routers. Egress peer TE is enabled on R3. The ingress Router R0 directs traffic destined to a remote network to Router R3, which has egress peer TE enabled.

Figure 58: Configuring Egress Peer Traffic Engineering Using BGP Labeled Unicast



Configuration

IN THIS SECTION

- CLI Quick Configuration | **888**
- Configuring Router R3 | **898**
- Results | **902**

CLI Quick Configuration

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

Router R0

```
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.1/24
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:10:1:1::1/64
set interfaces ge-2/1/4 unit 0 family inet address 10.100.100.1/30
set interfaces ge-2/1/4 unit 0 family inet6 address ::10.100.100.1/126
set interfaces ge-2/1/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.1.1.1/32
set interfaces lo0 unit 0 family inet6 address ::10.1.1.1/128
set routing-options router-id 01.1.1.1
set routing-options autonomous-system 64496
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls ipv6-tunneling
set protocols mpls no-cspf
set protocols mpls label-switched-path to_asbr1_r3 to 10.6.6.6
set protocols mpls label-switched-path to_asbr2_r4 to 10.7.7.7
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR-1-2 type internal
set protocols bgp group RR-1-2 local-address 10.1.1.1
set protocols bgp group RR-1-2 family inet unicast add-path receive
set protocols bgp group RR-1-2 family inet unicast add-path send path-count 6
set protocols bgp group RR-1-2 family inet labeled-unicast rib inet.3
set protocols bgp group RR-1-2 family inet6 unicast add-path receive
set protocols bgp group RR-1-2 family inet6 unicast add-path send path-count 6
set protocols bgp group RR-1-2 family inet6 labeled-unicast rib inet6.3
set protocols bgp group RR-1-2 export exp-svr-pre
set protocols bgp group RR-1-2 export nhs
set protocols bgp group RR-1-2 neighbor 10.4.4.4
set protocols bgp group R0RT0 type external
set protocols bgp group R0RT0 family inet unicast
set protocols bgp group R0RT0 peer-as  64496
set protocols bgp group R0RT0 neighbor 10.1.1.2
set protocols bgp group R0RT0-v6 type external
set protocols bgp group R0RT0-v6 family inet6 unicast
set protocols bgp group R0RT0-v6 peer-as  64496
set protocols bgp group R0RT0-v6 neighbor 2001:db8:40::2
set protocols ospf area 0.0.0.0 interface ge-2/1/4.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
```

```
set protocols ldp interface fxp0.0 disable
set policy-options prefix-list server_v4_prefix 10.1.1.1/32
set policy-options prefix-list server_v6_prefix ::10.1.1.1/128
set policy-options policy-statement exp-svr-pre term 1 from prefix-list server_v4_prefix
set policy-options policy-statement exp-svr-pre term 1 then accept
set policy-options policy-statement exp-svr-pre term 2 from prefix-list server_v6_prefix
set policy-options policy-statement exp-svr-pre term 2 then accept
set policy-options policy-statement nhs then next-hop self
```

Router R1

```
set interfaces ge-1/0/4 unit 0 family inet address 00.100.100.2/30
set interfaces ge-1/0/4 unit 0 family inet6 address ::00.100.100.2/126
set interfaces ge-1/0/4 unit 0 family mpls
set interfaces ge-1/0/6 unit 0 family inet address 10.100.104.1/30
set interfaces ge-1/0/6 unit 0 family inet6 address ::10.100.104.1/126
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces ge-1/0/9 unit 0 family inet address 100.100.105.1/30
set interfaces ge-1/0/9 unit 0 family inet6 address ::10.100.105.1/126
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces ge-1/1/3 unit 0 family inet address 10.100.102.1/30
set interfaces ge-1/1/3 unit 0 family inet6 address ::10.100.102.1/126
set interfaces ge-1/1/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.3.3.3/32
set interfaces lo0 unit 0 family inet6 address ::10.3.3.3/128
set routing-options router-id 10.3.3.3
set routing-options autonomous-system 64496
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls ipv6-tunneling
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 ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

Router R2

```
set interfaces et-0/0/0 unit 0 family inet address 10.100.102.2/30
set interfaces et-0/0/0 unit 0 family inet6 address ::10.100.102.2/126
set interfaces et-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.4.4.4/32
set interfaces lo0 unit 0 family inet6 address ::10.4.4.4/128
set routing-options router-id 10.4.4.4
set routing-options autonomous-system 64496
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls ipv6-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group Client type internal
set protocols bgp group Client local-address 10.4.4.4
set protocols bgp group Client advertise-inactive
set protocols bgp group Client family inet unicast add-path receive
set protocols bgp group Client family inet unicast add-path send path-count 6
set protocols bgp group Client family inet labeled-unicast rib inet.3
set protocols bgp group Client family inet6 unicast add-path receive
set protocols bgp group Client family inet6 unicast add-path send path-count 6
set protocols bgp group Client family inet6 labeled-unicast rib inet6.3
set protocols bgp group Client cluster 10.4.4.4
set protocols bgp group Client neighbor 10.1.1.1
set protocols bgp group Client neighbor 10.6.6.6
set protocols bgp group Client neighbor 10.7.7.7
set protocols ospf area 0.0.0.0 interface et-0/0/0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

Router R3

```
set interfaces ge-1/1/0 unit 0 family inet address 10.100.104.2/30
set interfaces ge-1/1/0 unit 0 family inet6 address ::10.100.104.2/126
set interfaces ge-1/1/0 unit 0 family mpls
set interfaces ge-2/2/5 unit 0 family inet address 10.200.203.1/28
set interfaces ge-2/2/5 unit 0 family inet6 address ::10.200.203.1/124
set interfaces ge-2/2/8 unit 0 family inet address 10.200.202.1/30
```

```
set interfaces ge-2/2/8 unit 0 family inet6 address ::10.200.202.1/126
set interfaces lo0 unit 0 family inet address 10.6.6.6/32
set interfaces lo0 unit 0 family inet6 address ::10.6.6.6/128
set routing-options router-id 10.6.6.6
set routing-options autonomous-system 64496
set routing-options forwarding-table export pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls ipv6-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp log-updown
set protocols bgp group RR-1-2 type internal
set protocols bgp group RR-1-2 local-address 10.6.6.6
set protocols bgp group RR-1-2 family inet unicast add-path receive
set protocols bgp group RR-1-2 family inet unicast add-path send path-count 6
set protocols bgp group RR-1-2 family inet labeled-unicast rib inet.3
set protocols bgp group RR-1-2 family inet6 unicast add-path receive
set protocols bgp group RR-1-2 family inet6 unicast add-path send path-count 6
set protocols bgp group RR-1-2 family inet6 labeled-unicast rib inet6.3
set protocols bgp group RR-1-2 export exp-arp-to-rrs
set protocols bgp group RR-1-2 neighbor 10.4.4.4
set protocols bgp group Peer1-lan-1 type external
set protocols bgp group Peer1-lan-1 family inet unicast
set protocols bgp group Peer1-lan-1 export exp_server_v4_v6_peers
set protocols bgp group Peer1-lan-1 peer-as 64497
set protocols bgp group Peer1-lan-1 neighbor 10.200.202.2 egress-te
set protocols bgp group Peer1-lan-1 neighbor 10.200.203.2 egress-te
set protocols bgp group Peer1-lan-1-v6 family inet6 unicast
set protocols bgp group Peer1-lan-1-v6 export exp_server_v4_v6_peers
set protocols bgp group Peer1-lan-1-v6 peer-as 64497
set protocols bgp group Peer1-lan-1-v6 neighbor ::10.200.202.2 egress-te
set protocols bgp group Peer1-lan-1-v6 neighbor ::10.200.203.2 egress-te
set protocols ospf area 0.0.0.0 interface ge-1/1/0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options prefix-list server_v4_pre 10.1.1.1/32
set policy-options prefix-list server_v6_pre ::10.1.1.1/128
set policy-options policy-statement exp-arp-to-rrs term 1 from protocol arp
set policy-options policy-statement exp-arp-to-rrs term 1 from rib inet.3
set policy-options policy-statement exp-arp-to-rrs term 1 then next-hop self
```

```
set policy-options policy-statement exp-arp-to-rrs term 1 then accept
set policy-options policy-statement exp-arp-to-rrs term 2 from protocol arp
set policy-options policy-statement exp-arp-to-rrs term 2 from rib inet6.3
set policy-options policy-statement exp-arp-to-rrs term 2 then next-hop self
set policy-options policy-statement exp-arp-to-rrs term 2 then accept
set policy-options policy-statement exp-arp-to-rrs term 3 from protocol bgp
set policy-options policy-statement exp-arp-to-rrs term 3 then accept
set policy-options policy-statement exp-arp-to-rrs term 4 then reject
set policy-options policy-statement exp_server_v4_v6_peers term 1 from prefix-list server_v4_pre
set policy-options policy-statement exp_server_v4_v6_peers term 1 then accept
set policy-options policy-statement exp_server_v4_v6_peers term 2 from prefix-list server_v6_pre
set policy-options policy-statement exp_server_v4_v6_peers term 2 then accept
set policy-options policy-statement pplb then load-balance per-packet
```

Router R4

```
set interfaces ge-3/0/2 vlan-tagging
set interfaces ge-3/0/2 unit 0 vlan-id 1
set interfaces ge-3/0/2 unit 0 family inet address 10.200.204.1/24
set interfaces ge-3/0/2 unit 0 family inet6 address ::10.200.204.1/120
set interfaces ge-3/0/2 unit 0 family mpls
set interfaces ge-3/0/2 unit 1 vlan-id 2
set interfaces ge-3/2/4 unit 0 family inet address 10.100.105.2/30
set interfaces ge-3/2/4 unit 0 family inet6 address ::10.100.105.2/126
set interfaces ge-3/2/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.7.7.7/32
set interfaces lo0 unit 0 family inet6 address ::10.7.7.7/128
set routing-options router-id 10.7.7.7
set routing-options autonomous-system 64496
set routing-options forwarding-table export pplb
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls ipv6-tunneling
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group RR-1-2 type internal
set protocols bgp group RR-1-2 local-address 10.7.7.7
set protocols bgp group RR-1-2 family inet unicast add-path receive
set protocols bgp group RR-1-2 family inet unicast add-path send path-count 6
set protocols bgp group RR-1-2 family inet labeled-unicast rib inet.3
set protocols bgp group RR-1-2 family inet6 unicast add-path receive
set protocols bgp group RR-1-2 family inet6 unicast add-path send path-count 6
```

```
set protocols bgp group RR-1-2 family inet6 labeled-unicast rib inet6.3
set protocols bgp group RR-1-2 export exp-arp-to-rrs
set protocols bgp group RR-1-2 neighbor 10.4.4.4
set protocols bgp group Peer5-6-lan type external
set protocols bgp group Peer5-6-lan family inet unicast
set protocols bgp group Peer5-6-lan export exp_server_v4_v6_peers
set protocols bgp group Peer5-6-lan peer-as 64497
set protocols bgp group Peer5-6-lan-v6 type external
set protocols bgp group Peer5-6-lan-v6 family inet6 unicast
set protocols bgp group Peer5-6-lan-v6 export exp_server_v4_v6_peers
set protocols bgp group Peer5-6-lan-v6 peer-as 64497
set protocols ospf area 0.0.0.0 interface ge-3/2/4.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options prefix-list server_v4_pre 10.1.1.1/32
set policy-options prefix-list server_v6_pre ::10.1.1.1/128
set policy-options policy-statement exp-arp-to-rrs term 1 from protocol arp
set policy-options policy-statement exp-arp-to-rrs term 1 from rib inet.3
set policy-options policy-statement exp-arp-to-rrs term 1 then next-hop self
set policy-options policy-statement exp-arp-to-rrs term 1 then accept
set policy-options policy-statement exp-arp-to-rrs term 2 from protocol arp
set policy-options policy-statement exp-arp-to-rrs term 2 from rib inet6.3
set policy-options policy-statement exp-arp-to-rrs term 2 then next-hop self
set policy-options policy-statement exp-arp-to-rrs term 2 then accept
set policy-options policy-statement exp-arp-to-rrs term 3 from protocol bgp
set policy-options policy-statement exp-arp-to-rrs term 3 then accept
set policy-options policy-statement exp-arp-to-rrs term 4 then reject
set policy-options policy-statement exp_server_v4_v6_peers term 1 from prefix-list server_v4_pre
set policy-options policy-statement exp_server_v4_v6_peers term 1 then accept
set policy-options policy-statement exp_server_v4_v6_peers term 2 from prefix-list server_v6_pre
set policy-options policy-statement exp_server_v4_v6_peers term 2 then accept
set policy-options policy-statement pplb then load-balance per-packet
```

Router R5

```
set interfaces ge-0/2/1 unit 0 family inet address 10.100.140.1/30
set interfaces ge-0/2/1 unit 0 family inet6 address ::10.100.140.1/126
set interfaces ge-0/3/1 unit 0 family inet address 10.200.203.2/28
set interfaces ge-0/3/1 unit 0 family inet6 address ::10.200.203.2/124
set interfaces ge-0/3/4 unit 0 family inet address 10.200.202.2/30
```

```
set interfaces ge-0/3/4 unit 0 family inet6 address ::10.200.202.2/126
set interfaces lo0 unit 0 family inet address 10.8.8.8/32
set interfaces lo0 unit 0 family inet6 address ::10.8.8.8/128
set routing-options router-id 10.8.8.8
set routing-options autonomous-system 64497
set protocols bgp group Peer1-lan-1 type external
set protocols bgp group Peer1-lan-1 family inet unicast
set protocols bgp group Peer1-lan-1 export exp-lo0
set protocols bgp group Peer1-lan-1 peer-as  64497
set protocols bgp group Peer1-lan-1 neighbor 10.200.202.1
set protocols bgp group Peer1-lan-1 neighbor 10.200.203.1
set protocols bgp group Peer1-lan-1-v6 family inet6 unicast
set protocols bgp group Peer1-lan-1-v6 export exp-lo0
set protocols bgp group Peer1-lan-1-v6 peer-as  64497
set protocols bgp group Peer1-lan-1-v6 neighbor ::10.200.202.1
set protocols bgp group Peer1-lan-1-v6 neighbor ::10.200.203.1
set protocols bgp group Peer1-H1 type external
set protocols bgp group Peer1-H1 family inet unicast
set protocols bgp group Peer1-H1 neighbor 10.100.140.2 peer-as 64498
set protocols bgp group Peer1-H1-v6 type external
set protocols bgp group Peer1-H1-v6 family inet6 unicast
set protocols bgp group Peer1-H1-v6 neighbor ::10.100.140.2 peer-as 64498
set policy-options policy-statement exp-lo0 term 1 from interface lo0.0
set policy-options policy-statement exp-lo0 term 1 then accept
```

Router R6

```
set interfaces ge-1/1/2 unit 0 family inet address 10.100.140.2/30
set interfaces ge-1/1/2 unit 0 family inet6 address ::10.100.140.2/126
set interfaces ge-1/1/5 unit 0 family inet address 10.1.1.1/24
set interfaces ge-1/1/5 unit 0 family inet6 address 2001:db8:10:1:1::1/120
set interfaces lo0 unit 0 family inet address 10.17.17.1/32
set interfaces lo0 unit 0 family inet address 10.17.17.2/32
set interfaces lo0 unit 0 family inet address 10.17.17.3/32
set interfaces lo0 unit 0 family inet address 10.17.17.4/32
set interfaces lo0 unit 0 family inet address 10.17.17.5/32
set interfaces lo0 unit 0 family inet address 10.17.17.6/32
set interfaces lo0 unit 0 family inet address 10.17.17.7/32
set interfaces lo0 unit 0 family inet address 10.17.17.8/32
set interfaces lo0 unit 0 family inet address 10.17.17.9/32
set interfaces lo0 unit 0 family inet6 address ::10.17.17.1/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.2/128
```

```
set interfaces lo0 unit 0 family inet6 address ::10.17.17.3/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.4/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.5/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.6/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.7/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.8/128
set interfaces lo0 unit 0 family inet6 address ::10.17.17.9/128
set routing-options router-id 10.17.17.1
set routing-options autonomous-system 64498
set protocols bgp group H1-Peer1 type external
set protocols bgp group H1-Peer1 family inet unicast
set protocols bgp group H1-Peer1 export exp-lo0
set protocols bgp group H1-Peer1 neighbor 10.100.140.1 peer-as 64497
set protocols bgp group H1-Peer1-v6 type external
set protocols bgp group H1-Peer1-v6 family inet6 unicast
set protocols bgp group H1-Peer1-v6 export exp-lo0
set protocols bgp group H1-Peer1-v6 neighbor ::100.100.140.1 peer-as 64497
set protocols bgp group R6RT0 type external
set protocols bgp group R6RT0 family inet unicast
set protocols bgp group R6RT0 peer-as 65300
set protocols bgp group R6RT0 neighbor 10.1.1.2
set protocols bgp group R6RT0-v6 type external
set protocols bgp group R6RT0-v6 family inet6 unicast
set protocols bgp group R6RT0-v6 peer-as 65300
set protocols bgp group R6RT0-v6 neighbor 2008:db8:50::2
set policy-options policy-statement exp-lo0 term 1 from interface lo0.0
set policy-options policy-statement exp-lo0 term 1 then accept
set policy-options policy-statement exp-lo0 term 2 from protocol direct
set policy-options policy-statement exp-lo0 term 2 from protocol local
set policy-options policy-statement exp-lo0 term 2 then accept
```

Router R7

```
set interfaces ge-1/0/6 unit 0 family inet address 10.100.141.1/30
set interfaces ge-1/0/6 unit 0 family inet6 address ::10.100.141.1/126
set interfaces ge-1/1/4 vlan-tagging
set interfaces ge-1/1/4 unit 0 vlan-id 1
set interfaces ge-1/1/4 unit 0 family inet address 10.200.204.2/24
set interfaces ge-1/1/4 unit 0 family inet6 address ::10.200.204.2/120
set interfaces ge-1/1/4 unit 1 vlan-id 2
set interfaces ge-1/1/4 unit 2 vlan-id 3
set interfaces lo0 unit 0 family inet address 10.9.9.9/32
```

```
set interfaces lo0 unit 0 family inet6 address ::10.9.9.9/128
set routing-options router-id 10.9.9.9
set routing-options autonomous-system 64497
set protocols bgp group Peer1-lan-1 type external
set protocols bgp group Peer1-lan-1 family inet unicast
set protocols bgp group Peer1-lan-1 export exp-lo0
set protocols bgp group Peer1-lan-1 peer-as  64497
set protocols bgp group Peer1-lan-1 neighbor 10.200.204.1
set protocols bgp group Peer1-lan-1-v6 family inet6 unicast
set protocols bgp group Peer1-lan-1-v6 export exp-lo0
set protocols bgp group Peer1-lan-1-v6 peer-as  64497
set protocols bgp group Peer1-lan-1-v6 neighbor ::10.200.204.1
set protocols bgp group Peer2-H2 type external
set protocols bgp group Peer2-H2 family inet unicast
set protocols bgp group Peer2-H2 neighbor 10.100.141.2 peer-as 64499
set protocols bgp group Peer2-H2-v6 type external
set protocols bgp group Peer2-H2-v6 family inet6 unicast
set protocols bgp group Peer2-H2-v6 neighbor ::10.100.141.2 peer-as 64499
set policy-options policy-statement exp-lo0 term 1 from interface lo0.0
set policy-options policy-statement exp-lo0 term 1 then accept
```

Router R8

```
set interfaces ge-4/0/5 unit 0 family inet address 10.100.141.2/30
set interfaces ge-4/0/5 unit 0 family inet6 address ::10.100.141.2/126
set interfaces lo0 unit 0 family inet address 10.18.18.1/32
set interfaces lo0 unit 0 family inet address 10.18.18.2/32
set interfaces lo0 unit 0 family inet address 10.18.18.3/32
set interfaces lo0 unit 0 family inet address 10.18.18.4/32
set interfaces lo0 unit 0 family inet address 10.18.18.5/32
set interfaces lo0 unit 0 family inet address 10.18.18.6/32
set interfaces lo0 unit 0 family inet address 10.18.18.7/32
set interfaces lo0 unit 0 family inet address 10.18.18.8/32
set interfaces lo0 unit 0 family inet address 10.18.18.9/32
set interfaces lo0 unit 0 family inet6 address ::10.18.18.1/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.2/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.3/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.4/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.5/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.6/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.7/128
set interfaces lo0 unit 0 family inet6 address ::10.18.18.8/128
```

```
set interfaces lo0 unit 0 family inet6 address ::10.18.18.9/128
set routing-options router-id 10.18.18.1
set routing-options autonomous-system 64499
set protocols bgp group H2-Peer2 type external
set protocols bgp group H2-Peer2 family inet unicast
set protocols bgp group H2-Peer2 export exp-lo0
set protocols bgp group H2-Peer2 neighbor 10.100.141.1 peer-as 64497
set protocols bgp group H2-Peer2-v6 type external
set protocols bgp group H2-Peer2-v6 family inet6 unicast
set protocols bgp group H2-Peer2-v6 export exp-lo0
set protocols bgp group H2-Peer2-v6 neighbor ::10.100.141.1 peer-as 64497
set protocols bgp group R8RT0 type external
set protocols bgp group R8RT0 family inet unicast
set protocols bgp group R8RT0 peer-as 65400
set protocols bgp group R8RT0 neighbor 10.1.1.2
set policy-options policy-statement exp-lo0 term 1 from interface lo0.0
set policy-options policy-statement exp-lo0 term 1 then accept
set policy-options policy-statement exp-lo0 term 2 then reject
```

Configuring Router R3

Step-by-Step Procedure

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

To configure Router R3:

NOTE: Repeat this procedure for other routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv4 and IPv6 addresses.

```
[edit interfaces]
user@R3# set ge-1/1/0 unit 0 family inet address 10.100.104.2/30
user@R3# set ge-1/1/0 unit 0 family inet6 address ::10.100.104.2/126
user@R3# set ge-1/1/0 unit 0 family mpls
user@R3# set ge-2/2/5 unit 0 family inet address 10.200.203.1/28
user@R3# set ge-2/2/5 unit 0 family inet6 address ::10.200.203.1/124
```

```
user@R3# set ge-2/2/8 unit 0 family inet address 10.200.202.1/30
user@R3# set ge-2/2/8 unit 0 family inet6 address ::10.200.202.1/126
```

2. Configure the loopback addresses.

```
[edit interfaces]
user@R3# set lo0 unit 0 family inet address 10.6.6.6/32
user@R3# set lo0 unit 0 family inet6 address ::10.6.6.6/128
```

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

```
[edit routing-options]
user@R3# set router-id 10.6.6.6
user@R3# set autonomous-system 64496
```

4. Configure the RSVP protocol for all interfaces except the management interface.

```
[edit protocols]
user@R3# set rsvp interface all
user@R3# set rsvp interface fxp0.0 disable
```

5. Configure the MPLS protocol for all interfaces except the management interface.

```
[edit protocols]
user@R3# set mpls ipv6-tunneling
user@R3# set mpls interface all
user@R3# set mpls interface fxp0.0 disable
```

6. Configure IBGP peering sessions on the core-facing interface.

```
[edit protocols]
user@R3# set bgp log-updown
user@R3# set bgp group RR-1-2 type internal
user@R3# set bgp group RR-1-2 local-address 10.6.6.6
user@R3# set bgp group RR-1-2 family inet unicast add-path receive
user@R3# set bgp group RR-1-2 family inet unicast add-path send path-count 6
user@R3# set bgp group RR-1-2 family inet labeled-unicast rib inet.3
user@R3# set bgp group RR-1-2 family inet6 unicast add-path receive
```

```
user@R3# set bgp group RR-1-2 family inet6 unicast add-path send path-count 6
user@R3# set bgp group RR-1-2 family inet6 labeled-unicast rib inet6.3
user@R3# set bgp group RR-1-2 neighbor 10.4.4.4
```

7. Configure EBGP peering sessions on interfaces facing external edge routers.

```
[edit protocols]
user@R3# set bgp group Peer1-lan-1 type external
user@R3# set bgp group Peer1-lan-1 family inet unicast
user@R3# set bgp group Peer1-lan-1 peer-as 64497
user@R3# set bgp group Peer1-lan-1-v6 family inet6 unicast
user@R3# set bgp group Peer1-lan-1-v6 peer-as 64497
```

8. Enable egress peer traffic engineering for external BGP group Peer1-lan-1 and for the IPv6 group Peer1-lan-1-v6.

```
[edit protocols]
user@R3# set bgp group Peer1-lan-1 neighbor 10.200.202.2 egress-te
user@R3# set bgp group Peer1-lan-1 neighbor 10.200.203.2 egress-te
user@R3# set bgp group Peer1-lan-1-v6 neighbor ::10.200.202.2 egress-te
user@R3# set bgp group Peer1-lan-1-v6 neighbor ::10.200.203.2 egress-te
```

9. Configure the OSPF protocol as the IGP.

```
[edit protocols]
user@R3# set ospf area 0.0.0.0 interface ge-1/1/0.0
user@R3# set ospf area 0.0.0.0 interface fxp0.0 disable
user@R3# set ospf area 0.0.0.0 interface lo0.0 passive
user@R3# set ldp interface all
user@R3# set ldp interface fxp0.0 disable
```

10. Define a policy for exporting ARP routes to route reflectors.

```
[edit policy-options]
user@R3# set policy-statement exp-arp-to-rrs term 1 from protocol arp
user@R3# set policy-statement exp-arp-to-rrs term 1 from rib inet.3
user@R3# set policy-statement exp-arp-to-rrs term 1 then next-hop self
user@R3# set policy-statement exp-arp-to-rrs term 1 then accept
user@R3# set policy-statement exp-arp-to-rrs term 2 from protocol arp
```

```
user@R3# set policy-statement exp-arp-to-rrs term 2 from rib inet6.3
user@R3# set policy-statement exp-arp-to-rrs term 2 then next-hop self
user@R3# set policy-statement exp-arp-to-rrs term 2 then accept
user@R3# set policy-statement exp-arp-to-rrs term 3 from protocol bgp
user@R3# set policy-statement exp-arp-to-rrs term 3 then accept
user@R3# set policy-statement exp-arp-to-rrs term 4 then reject
```

11. Apply the policy *exp-arp-to-rrs* for exporting ARP routes to route reflectors to the external BGP group, ebgp-v6.

```
[edit protocols]
user@R3# set bgp group RR-1-2 export exp-arp-to-rrs
```

12. Define prefix lists with IPv4 and IPv6 routes.

```
[edit policy-options]
user@R3# set prefix-list server_v4_pre 10.1.1.1/32
user@R3# set prefix-list server_v6_pre ::10.1.1.1/128
```

13. Define a policy to export IPv4 and IPv6 routes to the server.

```
[edit policy-options]
user@R3# set policy-statement exp_server_v4_v6_peers term 1 from prefix-list server_v4_pre
user@R3# set policy-statement exp_server_v4_v6_peers term 1 then accept
user@R3# set policy-statement exp_server_v4_v6_peers term 2 from prefix-list server_v6_pre
user@R3# set policy-statement exp_server_v4_v6_peers term 2 then accept
```

14. Apply the policy to export IPv4 and IPv6 peer routes.

```
[edit protocols]
user@R3# set bgp group Peer1-lan-1 export exp_server_v4_v6_peers
user@R3# set bgp group Peer1-lan-1-v6 export exp_server_v4_v6_peers
```

15. Define a per-packet load-balancing policy.

```
[edit policy-options]
user@R3# set policy-statement pplb then load-balance per-packet
```

16. Apply the per-packet load-balancing policy.

```
[edit routing-options]
user@R3# set forwarding-table export pplb
```

Results

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

```
[edit]
user@R3# show interfaces
ge-1/1/0 {
    unit 0 {
        family inet {
            address 10.100.104.2/30;
        }
        family inet6 {
            address ::10.100.104.2/126;
        }
        family mpls;
    }
}
ge-2/2/5 {
    unit 0 {
        family inet {
            address 100.200.203.1/28;
        }
        family inet6 {
            address ::10.200.203.1/124;
        }
    }
}
ge-2/2/8 {
    unit 0 {
        family inet {
            address 10.200.202.1/30;
        }
        family inet6 {
            address ::10.200.202.1/126;
```

```
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.6.6.6/32;
        }
        family inet6 {
            address ::10.6.6.6/128;
        }
    }
}
```

```
[edit]
user@R3# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    ipv6-tunneling;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    log-updown;
    group RR-1-2 {
        type internal;
        local-address 10.6.6.6;
        family inet {
            unicast {
                add-path {
                    receive;
                    send {
                        path-count 6;
                    }
```

```
                }
            }
            labeled-unicast {
                rib {
                    inet.3;
                }
            }
        }
        family inet6 {
            unicast {
                add-path {
                    receive;
                    send {
                        path-count 6;
                    }
                }
            }
            labeled-unicast {
                rib {
                    inet6.3;
                }
            }
        }
        export exp-arp-to-rrs;
        neighbor 10.4.4.4;
    }
    group Peer1-lan-1 {
        type external;
        family inet {
            unicast;
        }
        export exp_server_v4_v6_peers;
        peer-as 64497;
        neighbor 10.200.202.2 {
            egress-te;
        }
        neighbor 10.200.203.2 {
            egress-te;
        }
    }
    group Peer1-lan-1-v6 {
        family inet6 {
            unicast;
```

```
        }
        export exp_server_v4_v6_peers;
        peer-as 64497;
        neighbor ::10.200.202.2 {
            egress-te;
        }
        neighbor ::10.200.203.2 {
            egress-te;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/1/0.0;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

```
[edit]
user@R3# show routing-options
router-id 10.6.6.6;
autonomous-system 64496;
    forwarding-table {
        export pplb;
    }
```

```
[edit]
user@R3# show policy-options
    prefix-list server_v4_pre {
```

```
        10.1.1.1/32;
    }
    prefix-list server_v6_pre {
        ::10.1.1.1/128;
    }
    policy-statement exp-arp-to-rrs {
        term 1 {
            from {
                protocol arp;
                rib inet.3;
            }
            then {
                next-hop self;
                accept;
            }
        }
        term 2 {
            from {
                protocol arp;
                rib inet6.3;
            }
            then {
                next-hop self;
                accept;
            }
        }
        term 3 {
            from protocol bgp;
            then accept;
        }
        term 4 {
            then reject;
        }
    }
    policy-statement exp_server_v4_v6_peers {
        term 1 {
            from {
                prefix-list server_v4_pre;
            }
            then accept;
        }
        term 2 {
            from {
```

```
                prefix-list server_v6_pre;
            }
            then accept;
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
}
```

Verification

IN THIS SECTION

- Identifying the Label and the Protocol Next Hop | **907**
- Verifying the Path of Packet with Label 299888 | **909**
- Verifying That Egress Peer Traffic Engineering Is Enabled on Router R3 | **910**

Confirm that the configuration is working properly.

Identifying the Label and the Protocol Next Hop

Purpose

Get the label number of the packet transported from R0 to R6 and the next hop from the routing table for route 10.17.17.2.

Action

From operational mode, run the **show route 10.17.17.2 extensive active-path** command on Router R0.

```
user@R0> show route 10.17.17.2 extensive active-path
inet.0: 262 destinations, 516 routes (261 active, 0 holddown, 1 hidden)
10.17.17.1/32 (3 entries, 1 announced)
TSI:
```

```
KRT in-kernel 10.17.17.1/32 -> {indirect(1048576)}
Page 0 idx 0, (group R0RT0 type External) Type 1 val 0x9a87fe0 (adv_entry)
   Advertised metrics:
     Nexthop: Self
     AS path: [65100] 1 65010 I
     Communities:
Path 10.17.17.1 from 10.4.4.4 Vector len 4.  Val: 0
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x97724a0
                Next-hop reference count: 339
                Source: 10.4.4.4
                Next hop type: Router, Next hop index: 624
                Next hop: 10.100.100.2 via ge-2/1/4.0, selected
                Label-switched-path to_asbr1_r3
                Label operation: Push 299888, Push 300128(top)
                Label TTL action: prop-ttl, prop-ttl(top)
                Load balance label: Label 299888: None; Label 300128: None;
                Session Id: 0x145
                Protocol next hop: 10.200.201.2
                Indirect next hop: 0x9a4c550 1048576 INH Session ID: 0x148
                State: <Active Int Ext>
                Local AS:   65100 Peer AS:   65100
                              Age: 1:33       Metric2: 2
                Validation State: unverified
                Task: BGP_100.10.4.4.4+179
                Announcement bits (3): 0-KRT 5-BGP_RT_Background 6-Resolve tree 2
                AS path: 1 10 I (Originator)
                Cluster list:  10.4.4.4
                Originator ID: 10.6.6.6
                Accepted
                Localpref: 100
                Router ID: 10.4.4.4
                Addpath Path ID: 1
                Indirect next hops: 1
                         Protocol next hop: 10.200.202.2 Metric: 2
                         Indirect next hop: 0x9a4c550 1048576 INH Session ID: 0x148
                         Indirect path forwarding next hops: 1
                                 Next hop type: Router
                                 Next hop: 10.100.100.2 via ge-2/1/4.0
                                 Session Id: 0x145
                         10.200.201.2/32 Originating RIB: inet.3
                           Metric: 2                     Node path count: 1
```

```
                          Indirect nexthops: 1
                                Protocol Nexthop: 10.6.6.6 Metric: 2 Push 299888
                                Indirect nexthop: 0x9a4c220 - INH Session ID:
0x0                              Indirect path forwarding nexthops:
1                                                                    Nexthop: 100.100.100.2
via ge-2/1/4.0
```

Meaning

Both the packet label 299888 and the next hop 10.200.202.2 are displayed in the output.

Verifying the Path of Packet with Label 299888

Purpose

Trace the path of the label 299888 and verify that the VPN entry is present in the mpls.0 routing table.

Action

```
user@R3> show route table mpls.0 protocol vpn active-path label 299888 detail
mpls.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
523440 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 640
                Address: 0xecfa130
                Next-hop reference count: 2
                Next hop: 10.200.202.2 via ge-2/2/8.0, selected
                Label operation: Pop
                Load balance label: None;
                Session Id: 0x16f
                State: <Active Int Ext>
                             Local AS:   64496
                             Age: 3:49:16
                               Validation State: unverified
                               Task: BGP_RT_Background
                 Announcement bits (1): 1-KRT
                               AS path: I
                           Ref Cnt: 1
```

Meaning

The label 299888 with VPN entry and next hop 10.200.202.2 is present in the mpls.0 routing table.

Verifying That Egress Peer Traffic Engineering Is Enabled on Router R3

Purpose

Verify that the egress peer traffic engineering is configured on Router R3.

Action

```
user@R3> show route protocol arp detail match-prefix 10.200.202.2
inet.0: 263 destinations, 514 routes (262 active, 0 holddown, 1 hidden)

inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
10.200.201.2/32 (1 entry, 1 announced)
        *ARP    Preference: 170
                Next hop type: Router
                Address: 0xecf91e0
                Next-hop reference count: 5
                Next hop: 10.200.202.2 via ge-2/2/8.0, selected
                Label operation: Pop
                Load balance label: None;
                Session Id: 0x0
                State: <Active Int Ext>
                              Local AS:   64496
                                Age: 3:52:52
                                Validation State: unverified
                                Task: BgpEgressPeeringTE
                                Announcement bits (3): 2-Resolve tree 1 3-BGP_RT_Background 4-
Resolve tree 2
```

Meaning

The output indicates that BGP egress peer traffic engineering is enabled on Router R3.

SEE ALSO

egress-te

egress-te-backup-paths

Configuring Egress Peer Traffic Engineering by Using BGP Labeled Unicast and Enabling MPLS Fast Reroute | 883

Egress Peer Traffic Engineering Using BGP Labeled Unicast Overview | 882

Segment Routing Traffic Engineering at BGP Ingress Peer Overview

IN THIS SECTION

- Understanding Segment Routing Policies | 911
- BGP's Role in Route Selection from a Segment Routing Policy | 912
- Statically Configured Segment Routing Policies | 912
- Supported and Unsupported Features | 913

This feature enables BGP to support a segment routing policy for traffic engineering at ingress routers. The controller can specify a segment routing policy consisting of multiple paths to steer labeled or IP traffic. The segment routing policy adds an ordered list of segments to the header of a packet for traffic steering. BGP installs the candidate routes of the segment routing policy into routing tables bgp.inetcolor.0 or bgp.inet6color.0. BGP selects one route from the candidate routes for a particular segment routing traffic engineering policy, and installs it in the new routing tables inetcolor.0 or inet6color.0. This feature supports both statically configured as well as BGP-installed segment routing traffic engineering policies in the forwarding table at ingress routers.

Understanding Segment Routing Policies

In segment routing the controller allows the ingress nodes in a core network to steer traffic through explicit paths while eliminating the state for the explicit paths in intermediate nodes. An ordered list of segments associated with the segment routing policy is added to the header of a data packet. These segment lists or lists of segment identifiers (SIDs) represent paths in the network, which are the best candidate paths selected from multiple candidate paths learned from various sources. An ordered list of segments is encoded as a stack of labels. This feature enables steering a packet toward a specific path depending on the network or customer requirements. The traffic can be labeled or IP traffic and is steered with a label swap or a destination-based lookup toward these segment routing traffic engineering paths. You can configure static policies at ingress routers to steer traffic even when the link to the controller fails. Static segment routing policies are useful to ensure traffic steering when the controller is down or unreachable.

BGP's Role in Route Selection from a Segment Routing Policy

When BGP receives an update for segment routing traffic engineering subsequent address family identifier (SAFI) from the controller, BGP performs some basic checks and validation on these updates. Segments that are not MPLS labels are considered invalid. If the updates are valid then BGP installs the segment routing traffic engineering policy in the routing tables bgp.inetcolor.0 and bgp.inet6color.0 and these are subsequently installed in the routing tables inetcolor.0 or inet6color.0. These routing tables use attributes such as *distinguisher*, *endpoint address*, and *color* as the key.

Starting in Junos OS Release 20.2R1, Junos OS provides support for controller based BGP-SRTE routes are installed as segment routing traffic-engineered (SPRING-TE) routes. BGP installs the segment routing traffic engineering policy in the routing tables bgp.inetcolor.0 and bgp.inet6color.0 and these are subsequently installed in the routing tables inetcolor.0 or inet6color.0 by SPRING-TE.

The policy action `color: color-mode:color-value` is configured at the `[edit policy-options community name members]` hierarchy level to attach color communities when exporting prefixes from inet-unicast and inet6-unicast address families.

To enable BGP IPv4 segment routing traffic engineering capability for an address family, include the `segment-routing-te` statement at the `[edit protocols bgp family inet]` hierarchy level.

To enable BGP IPv6 segment routing traffic engineering capability for an address family include the `segment-routing-te` statement at the `[edit protocols bgp family inet6]` hierarchy level.

NOTE: Starting in Release 18.3R1, Junos OS supports collection of traffic statistics for both ingress IP and transit MPLS traffic in a network configured with segment routing traffic engineering policy. To enable collection of traffic statistics include the `telemetry` statement at the `[edit protocols source-packet-routing]` hierarchy level.

Statically Configured Segment Routing Policies

Static policies can be configured at ingress routers to allow routing of traffic even when the link to the controller fails. Configure `sr-preference` at the `[edit protocols source-packet-routing]` hierarchy level to choose a statically configured segment routing traffic engineering policy forwarding entry over a BGP-signaled segment routing traffic engineering forwarding entry. The top label of the segment identifier label stack is swapped with the interior gateway protocol (IGP) top label for resolution.

A static segment routing traffic engineering policy can contain multiple paths with or without weighted ECMP. If IGP configuration has weighted ECMP configured, then the forwarding path provides hierarchical weighted equal-cost multipath (ECMP). However, if weighted ECMP is not configured, equal balance is applied to all the segment routing traffic engineering paths.

Supported and Unsupported Features

Junos OS supports the following features with BGP segment routing traffic engineering:

- For PTX Series, this feature is supported for FPC-PTX-P1-A with enhanced chassis mode.
- Weighted ECMP and hierarchical weighted ECMP.
- MPLS fast reroute (FRR) is supported for the paths in segment routing traffic engineering policies. IGP backup paths corresponding to the top label are installed to the routing table when available for segment routing traffic engineering policy paths.

The following limitations apply to BGP segment routing traffic engineering::

- BGP and static segment routing traffic engineering policies are only supported for the master instance.
- The segment routing traffic engineering paths that are explicitly configured using static policies or learned through BGP are limited to lists of segment identifiers that represent absolute MPLS labels only.
- A maximum of 128 segment lists are supported for static segment routing traffic engineering policies.
- The BGP segment routing traffic engineering SAFI is not supported for peers in routing instances.
- The BGP segment routing traffic engineering network layer reachability information (NLRI) cannot be imported to other routing tables using routing information base (RIB) groups (RIBs are also known as routing tables).
- Traffic statistics are not supported for traffic traversing the segment routing policy.
- The processing of time-to-live (TTL) MPLS label segment identifiers is not supported.
- Nonstop active routing is not supported.
- Class-of-service (CoS) policies work on the top label.
- Only non-VPN CoS rewrite CLI commands are supported; for example, EXP rewrite for the top label is supported.
- For an ingress packet, a maximum of eight labels can be parsed, and Layer 2 or Layer 3 MPLS payload fields are used in the load-balancing hash calculation. If label depth in the ingress packet is more than eight labels, then MPLS payload is not parsed and Layer 2 and Layer 3 MPLS payload fields are not used in the load-balancing hash calculation.
- The maximum label stack depth support is five. You must configure `maximum-labels` to limit the label depth of segment routing traffic engineering policies. If `maximum-labels` is not configured, meaningful defaults apply that restrict the maximum label depth to five.

- The color attribute must be specified in segment routing traffic engineering LSP configuration. Hence the ingress routes are downloaded to inetcolor{6}.0 tables.
- When there are multiple static segment routing traffic engineering policies with the same `Endpoint`, `color` preference but different binding segment identifiers are present, the route corresponding to the lesser binding segment identifier is installed in the `mpls.0` table.
- Mixed segment identifiers are not supported: the segment identifiers in the segment routing traffic engineering segment list must be exclusively IPv4 or IPv6.
- You must explicitly configure MPLS maximum-labels on an interface to accommodate more than five labels; otherwise more than five labels might result in packet drops.
- The default limits of the supported parameters are listed below in Table 6 on page 914:

Table 6: Supported Parameters for Segment Routing Traffic Engineering

Parameter	Limit
Maximum number of labels supported	5
Maximum number of paths in segment routing traffic engineering policy	8
Number of BGP segment routing traffic engineering policies	32,000
Number of static segment routing traffic engineering policies	32,000

SEE ALSO

extended-nexthop-color

segment-list

source-routing-path

source-packet-routing

sr-preference-override

Configuring Ingress Traffic Engineering with Segment Routing in a BGP Network

Starting in Junos OS Release 17.4R1, a BGP speaker supports traffic steering based on a segment routing policy. The controller can specify a segment routing policy consisting of multiple paths to steer labeled or IP traffic. This feature enables BGP to support a segment routing policy for traffic engineering at ingress routers. The segment routing policy adds an ordered list of segments to the header of a packet for traffic steering. Static policies can be configured at ingress routers to allow routing of traffic even when the link to the controller fails.

NOTE: This feature is supported on PTX Series with FPC-PTX-P1-A. For devices that have multiple FPCs, you must configure enhanced mode on the chassis.

Before you begin configuring BGP to receive segment routing traffic engineering policy from the controller, do the following tasks:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and segment routing labels..
4. Configure BGP.
5. Configure segment routing on the controller and all other routers.

To configure traffic engineering for BGP segment routing:

1. Enable BGP IPv4 segment routing traffic engineering capability for an address family. This feature is available only for inet, inet unicast, inet6, and inet6 unicast network layer reachability information (NLRI) families.

```
[edit protocols bgp family name]
user@host# set segment-routing-te
```

For example, enable segment routing for a particular BGP group as follows:

```
[edit protocols bgp group srte]
user@host# set family inet
user@host# set family inet unicast
user@host# set family inet segment-routing-te
user@host# set family inet6 unicast
```

```
user@host# set family inet6 segment-routing-te
user@host# set neighbor 27.2.1.2
user@host# set neighbor 27.2.1.2 peer-as-600
```

2. Configure segment routing global block (SRGB). Junos OS uses this label block for steering the packets to a remote destination. Configure the start label and SRGB index range.

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label start-label-value
user@host# set srgb index-rangeindex-range-value
```

For example, configure the start label and the SRGB index range with the following values:

```
[edit protocols isis source-packet-routing]
user@host# set srgb start-label 800000
user@host# set protocols isis source-packet-routing srgb index-range 80000
```

3. Configure the policy action to attach color communities when exporting prefixes from inet-unicast and inet6-unicast address families.

```
[edit policy-options community name ]
user@host# set members color: color-mode: color-value
```

For example, configure the following color attributes for a BGP community:

```
[edit policy-options community srte_community ]
user@host# set members color: 2: 1200
```

4. Configure the source routing LSP for steering traffic at the ingress router. Specify the attributes such as the tunnel endpoint, color, binding segment identifier, and preference for traffic engineering. Configuring binding segment identifier installs the route in the MPLS tables.

```
[edit protocols source-packet-routing]
user@host# set source-routing-path name to to
user@host# set source-routing-path name color color
user@host# set source-routing-path name binding-sid binding-sid
user@host# set source-routing-path name preference preference
```

For example, you can configure the attributes as follows:

```
[edit protocols source-packet-routing]
user@host# set source-routing-path srtelsp1 to 7.7.7.7
user@host# set source-routing-path srtelsp1 color 1200
user@host# set source-routing-path srtelsp1 binding-sid 1200
user@host# set source-routing-path srtelsp1 preference 70
```

5. Configure weighted ECMP for the primary segment list of a segment routing path. If the forwarding interface is also configured with weighted ECMP then Junos OS applies hierarchical weighted ECMP. If you do not configure the weight percentage, then only IGP weights are applied on the forwarding interfaces.

```
[edit protocols source-packet-routing]
user@host# set source-routing-path name primary name weight weight
user@host# set source-routing-path name primary name weight weight
```

For example, you can configure the routing paths and weights as follows:

```
[edit protocols source-packet-routing]
user@host# set source-routing-path srtelsp1 primary sr1 weight 1
user@host# set source-routing-path srtelsp1 primary sr4 weight 2
```

6. Configure the segment routing preference for routes received for this tunnel. This segment routing preference value overrides the global segment routing preference value and is used to select between candidate segment routing policies installed by different protocols such as static and BGP.

```
[edit protocols source-packet-routing]
user@host# set sr-preference-override sr-preference-override
user@host# set sr-preference sr-preference
```

For example, you can configure the sr preference as follows:

```
[edit protocols source-packet-routing]
user@host# set sr-preference-override 300
user@host# set sr-preference 200
```

7. Configure static policies at ingress routers to allow routing of traffic even when the link to the controller fails. Specify one or more nexthop labels. The successfully resolved LSPs are used to resolve BGP payload prefixes that have the same color and endpoint.

```
[edit protocols source-packet-routing]
user@host# set segment-list segment-list-name hop-namelabel label
```

For example, configure two segment lists *sr1*, *sr4* and specify labels for steering segment routing traffic at an ingress router as follows:

```
[edit protocols source-packet-routing]
user@host# set segment-list sr1 hop1 label 801001
user@host# set segment-list sr1 hop2 label 801002
user@host# set segment-list sr1 hop3 label 801003
user@host# set segment-list sr1 hop4 label 801007
user@host# set segment-list sr4 hop1 label 801004
user@host# set segment-list sr4 hop2 label 801005
```

NOTE: If BGP and static segment routing are configured together for traffic engineering, then by default Junos OS chooses statically configured segment routing policies.

8. Configure segment routing preference overide to replace the received segment routing traffic engineering preference value with the configured override value. Segment routing policy preference can change based on certain tie-breaking rules involving sr-preference-override, sr-preference, and admin-preference.

```
[edit protocols bgp]
user@host# set sr-preference-override sr-preference-override
```

For example, configure the following value for BGP segment routing preference override:

```
[edit protocols bgp]
user@host# set sr-preference-override 400
```

SEE ALSO

- extended-nexthop-color
- *segment-list*
- *source-packet-routing*
- *source-routing-path*
- sr-preference-override
- Segment Routing Traffic Engineering at BGP Ingress Peer Overview | **911**

Enabling Traffic Statistics Collection for BGP Labeled Unicast

Starting in Junos OS Release 18.1R1, you can enable traffic statistics collection for BGP labeled unicast traffic at the ingress router in a network configured with segment routing. Traffic statistics are collected based on the label stack. For example, if there are two routes with the same label stack but different next-hops then traffic statistics are aggregated for these routes because the label stack is the same. Traffic statistics can be periodically collected and saved to a specified file based on the label stack received in the BGP route update. By default, traffic statistics collection is disabled. Enabling traffic statistics collection triggers a BGP import policy. Traffic statistics collection is supported only for IPv4 and IPv6 address families.

Before you begin configuring BGP to collect traffic statistics, do the following tasks:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and LDP.
4. Configure BGP.
5. Configure segment routing on the controller and all other routers.

In a network configured with segment routing, each node and link is assigned a segment identifier (SID), which is advertised through IGP or BGP. In an MPLS network, each segment is assigned a unique segment label that serves as the SID for that segment. Each forwarding path is represented as a segment routing label-switched path (LSP). The segment routing LSP is represented with a stack of SID labels at ingress. The ingress router can impose these labels to route the traffic. With BGP labeled unicast a controller can program the ingress router to steer traffic and advertise a prefix with a label stack.

To enable traffic statistics collection for BGP labeled unicast at ingress:

1. Enable collection of traffic statistics of labeled unicast IPv4 and IPv6 families for specific BGP groups or BGP neighbors.

```
[edit protocols bgp group name family inet labeled-unicast traffic statistics]
user@host# set labeled-path
```

2. Configure periodic traffic statistics collection for BGP label-switched paths in a segmented routing network and save the statistics to a file.

```
[edit protocols bgp]
user@host# set traffic-statistics-labeled-path
```

   a. Specify the filename to save the collected traffic statistics collected at a specified time interval.

```
[edit protocols bgp traffic-statistics-labeled-path]
user@host# set file filename
```

   b. Specify the time interval in seconds for collecting traffic statistics. You can specify a number from 60 to 65535 seconds.

```
[edit protocols bgp traffic-statistics-labeled-path]
user@host# set interval interval
```

SEE ALSO

traffic-statistics-labeled-path

show bgp group traffic-statistics

Understanding SRv6 Network Programming and Layer 3 Services over SRv6 in BGP

IN THIS SECTION

- Benefits of SRv6 Network Programming | 921
- SRv6 Network Programming in BGP Networks | 921
- Layer 3 VPN Services over the SRv6 Core | 922
- Advertising Layer 3 VPN Services to BGP Peers | 923
- Supported and Unsupported Features for SRv6 Network Programming in BGP | 923

Benefits of SRv6 Network Programming

- BGP leverages the segment routing capability of devices to set up Layer 3 VPN tunnels. IPv4 packets can be transported through an SRv6 ingress node even if the transit routers are not SRv6-capable. This eliminates the need to deploy segment routing on all nodes in an IPv6 network.
- Network programming depends entirely on the IPv6 header and the header extension to transport a packet, eliminating the need for protocols such as MPLS. This ensures a seamless deployment without any major hardware or software upgrade in a core IPv6 network.
- Junos OS supports all function behaviors on a single segment identifier (SID) and can inter-operate in both insert mode and encapsulation mode. This allows a single device to simultaneously play the provider (P) router and the provider edge (PE) router roles.

SRv6 Network Programming in BGP Networks

Network programming is the capability of a network to encode a network program into individual instructions that are inserted into the IPv6 packet headers. The Segment Routing Header (SRH) is a type of IPv6 routing extension header that contains a segment list encoded as an SRv6 SID. An SRv6 SID consists of the locator, which is an IPv6 address, and a function that defines a particular task for each SRv6-capable node in the SRv6 network. SRv6 network programming eliminates the need for MPLS and provides flexibility to leverage segment routing.

NOTE: Ensure that you use a unique SID, which BGP uses to allocate an SRv6 SID.

To configure IPv4 transport over the SRv6 core, include the `end-dt4-sid` *`sid`* statement at the `[edit protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level.

To configure IPv6 transport over the SRv6 core, include the `end-dt6-sid` *`sid`* statement at the `[edit routing protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level.

To configure IPv4 and IPv6 transport over the SRv6 core, include the `end-dt46-sid` *`sid`* statement at the `[edit routing protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level. The end-dt4-sid statement denotes the endpoint SID with de-encapsulation and IPv4 table lookup. The end dt6-sid statement is the endpoint with de-encapsulation and IPv6 table lookup. The end-dt46-sid statement is the endpoint with decapsulation and specific IP table lookup. The end-dt46 is a variant of end.dt4 and end.dt6 behavior. BGP allocates these values for IPv4 and IPv6 Layer3 VPN service SIDs.

Layer 3 VPN Services over the SRv6 Core

When connecting to the egress PE, the ingress PE encapsulates the payload in an outer IPv6 header where the destination address is the SRv6 service SID associated with the related BGP route update. The egress PE sets the next hop to one of its IPv6 addresses that is also the SRv6 locator from which the SRv6 service SID is allocated. Multiple routes can resolve through the same segment routing policy.

Figure 59: SRv6 Packet Encapsulation

Starting in Junos OS Release 20.4R1, you can configure BGP-based Layer 3 service over the SRv6 core. You can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the dataplane. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS. Such networks depend only on the IPv6 headers and header extensions for transmitting data.

NOTE: Ensure that the `end-dt4-sid` *`sid`* and the `end-dt6-sid` *`sid`* are the last SIDs in the segment list, or the destination address of the packet with no SRH header.

To configure IPv4 VPN services over the SRv6 core, include the `end-dt4-sid` statement at the `[edit routing-instances` *`instance-name`* `protocols bgp source-packet-routing srv6 locator` *`name`*`]` hierarchy level.

The end dt46 SID must be the last segment in a segment routing policy, and a SID instance must be associated with an IPv4 FIB table and an IPv6 FIB table.

Advertising Layer 3 VPN Services to BGP Peers

BGP advertises the reachability of prefixes of a particular service from an egress PE device to ingress PE nodes. BGP messages exchanged between PE devices carry SRv6 service SIDs, which BGP uses to interconnect PE devices to form VPN sessions. For Layer 3 VPN services where BGP uses a per-VRF SID allocation, the same SID is shared across multiple network layer reachability information (NLRI) address families.

To advertise SRv6 services to BGP peers at the egress node, include the `advertise-srv6-service` statement at the `[edit protocols bgp family inet6 unicast]` hierarchy level.

Egress PE devices that support SRv6-based Layer 3 services advertise overlay service prefixes along with a service SID. The BGP ingress node receives these advertisements and adds the prefix to the corresponding virtual routing and forwarding (VRF) table.

To accept SRv6 services at the ingress node, include the `accept-srv6-service` statement at the `[edit protocols bgp family inet6 unicast]` hierarchy level.

Supported and Unsupported Features for SRv6 Network Programming in BGP

Junos OS supports the following features with SRv6 Network Programming in BGP:

- Ingress devices support seven SIDs in the reduced mode including the VPN SID
- Egress devices support seven SIDs including the VPN SID
- Endpoint with de-encapsulation and specific IP table lookup (End.DT46 SID)

Junos OS does not support the following features in conjunction with SRv6 Network Programming in BGP:

- Fragmentation and reassembly in SRv6 tunnels
- VPN options B and C
- Detection of duplicate SIDs

SEE ALSO

srv6

advertise-srv6-service

accept-srv6-service

Example: Configuring Layer 3 Services over SRv6 in BGP Networks

IN THIS SECTION

- Requirements | 924
- Overview | 924
- Configuration | 926
- Verification | 942

This example shows how to configure SRv6 network programming and Layer 3 VPN services in BGP Networks. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS. This feature is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers with MPC7E, MPC8E, or MPC9E line cards
- Junos OS Release 20.4R1 or later

Overview

IN THIS SECTION

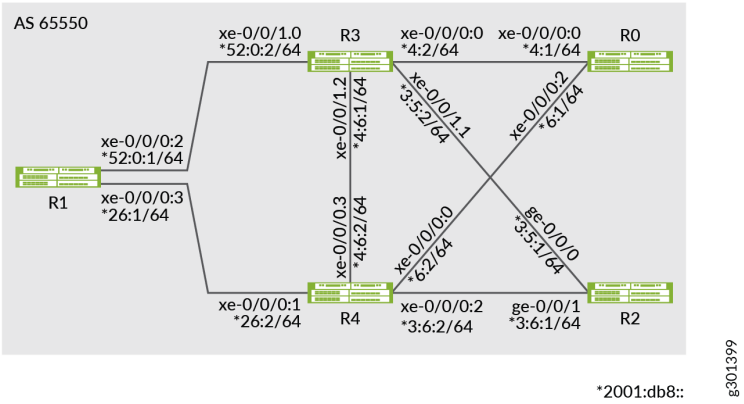
- Topology | 925

Starting in Junos OS Release 20.4R1, you can configure BGP-based Layer 3 services over the SRv6 core network. With SRv6 network programming, networks depend only on the IPv6 headers and header extensions for transmitting data. You can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the dataplane.

Topology

In Figure 60 on page 925, Router R0 is the ingress and Router R1 and R2 are the egress routers that support IPv4-only customer edge devices. Routers R3 and R4 comprise an IPv6-only provider core network. All routers belong to the same autonomous system. IS-IS is the interior gateway protocol configured to support SRv6 in the IPv6 core routers R3 and R4. In this example, BGP is configured on routers R0, R1, and R2. Router R0 is configured as an IPv6 route reflector with IBGP peering sessions to both Router R1 and Router R2. The egress Router R1 advertises the L3VPN SID to ingress Router R0, which accepts and updates the VRF table.

Figure 60: Layer 3 Services over SRv6 in BGP Networks



R1 is configured with 3011::1 as end-sid and all the BGP routes are advertised with 3011::1 as next hop to Router R0. Router R0 has two paths to R1, the primary path through R3 and the backup path through R4. In Router R0 , the primary path is with default metric and the backup path is configured with metric 50. Here are some of the routes that are advertised from Router R1 to R0:

IPv4	21.0.0.0
IPv6	2001:21::
IPv4 VPN	31.0.0.0
IPv6 VPN	2001:31::

Configuration

IN THIS SECTION

- CLI Quick Configuration | 926
- Configure Router R0 | 933
- Results | 937

CLI Quick Configuration

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

Router R0

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.1/30
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:1/64

set interfaces xe-0/0/0:2 unit 0 family inet address 1.6.1.1/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::0/128
set policy-options policy-statement adv_global term v4 from route-filter 20.0.0.0/8 orlonger
set policy-options policy-statement adv_global term v4 then next-hop self
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:20::/64 orlonger
set policy-options policy-statement adv_global term v6 then next-hop self
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set policy-options community vpn2-target members target:100:2
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 11.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1002
```

```
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 11.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:11:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1002
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:11:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3001::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3001::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface xe-0/0/0:3.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3001::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options router-id 128.53.38.52
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-PE-all type internal
set protocols bgp group to-PE-all local-address abcd::128:53:38:52
set protocols bgp group to-PE-all family inet unicast extended-nexthop
set protocols bgp group to-PE-all family inet unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet unicast accept-srv6-service
set protocols bgp group to-PE-all family inet-vpn unicast extended-nexthop
set protocols bgp group to-PE-all family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet-vpn unicast accept-srv6-service
set protocols bgp group to-PE-all family inet6 unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet6 unicast accept-srv6-service
set protocols bgp group to-PE-all family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to-PE-all family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-PE-all export adv_global
set protocols bgp group to-PE-all cluster 128.53.38.52
set protocols bgp group to-PE-all neighbor abcd::128:53:35:39
set protocols bgp group to-PE-all neighbor abcd::128:53:35:35
set protocols bgp group to-TG-global-v4 type external
set protocols bgp group to-TG-global-v4 local-address 11.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1001
set protocols bgp group to-TG-global-v4 neighbor 11.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:11:1:1::1
```

```
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1001
set protocols bgp group to-TG-global-v6 neighbor 2001:11:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3001::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3001::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis source-packet-routing srv6 locator loc1 end-sid 3001::1 flavor usd
set protocols isis level 1 disable
```

Router R1

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:2 unit 0 family inet address 2.5.1.1/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::52:0:1/64
set interfaces xe-0/0/0:3 unit 0 family inet address 2.6.1.1/30
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8::26:1/64
set policy-options policy-statement adv_global term v4 from route-filter 21.0.0.0/8 orlonger
set policy-options policy-statement adv_global term v4 from route-filter 12.1.1.1/30 orlonger
set policy-options policy-statement adv_global term v4 then next-hop 3011::1
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:21::/64 orlonger
set policy-options policy-statement adv_global term v6 from route-filter 2001:12:1:1::1/126
orlonger
set policy-options policy-statement adv_global term v6 then next-hop 3011::1
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement adv_vpn1 term v4 from route-filter 31.0.0.0/8 orlonger
set policy-options policy-statement adv_vpn1 term v4 from route-filter 12.1.1.5/30 orlonger
set policy-options policy-statement adv_vpn1 term v4 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v4 then next-hop 3011::1
set policy-options policy-statement adv_vpn1 term v4 then accept
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:31::/64 orlonger
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:12:1:1::5/126
orlonger
set policy-options policy-statement adv_vpn1 term v6 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v6 then next-hop 3011::1
set policy-options policy-statement adv_vpn1 term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set policy-options community vpn2-target members target:100:2
```

```
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 12.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1012
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 12.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:12:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1012
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:12:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3011::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3011::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface xe-0/0/1:0.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-export adv_vpn1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3011::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 next-hop 3001::1
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 resolve
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 next-hop 3001::1
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 resolve
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address abcd::128:53:35:39
set protocols bgp group to-RR family inet unicast extended-nexthop
set protocols bgp group to-RR family inet unicast advertise-srv6-service
set protocols bgp group to-RR family inet unicast accept-srv6-service
set protocols bgp group to-RR family inet-vpn unicast extended-nexthop
set protocols bgp group to-RR family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet-vpn unicast accept-srv6-service
set protocols bgp group to-RR family inet6 unicast advertise-srv6-service
set protocols bgp group to-RR family inet6 unicast accept-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-RR export adv_global
set protocols bgp group to-RR neighbor abcd::128:53:38:52
set protocols bgp group to-TG-global-v4 type external
```

```
set protocols bgp group to-TG-global-v4 local-address 12.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1011
set protocols bgp group to-TG-global-v4 neighbor 12.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:12:1:1::1
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1011
set protocols bgp group to-TG-global-v6 neighbor 2001:12:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3011::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3011::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis source-packet-routing srv6 locator loc1 end-sid 3011::1 flavor usd
set protocols isis level 1 disable
```

Router R2

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 unit 0 family inet address 3.5.1.1/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8::3:5:1/64
set interfaces ge-0/0/1 unit 0 family inet address 3.6.1.1/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family inet6 address 2001:db8::3:6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::2/128
set policy-options policy-statement adv_global term v4 from route-filter 22.0.0.0/8 orlonger
set policy-options policy-statement adv_global term v4 from route-filter 13.1.1.1/30 orlonger
set policy-options policy-statement adv_global term v4 then next-hop 3021::1
set policy-options policy-statement adv_global term v4 then accept
set policy-options policy-statement adv_global term v6 from route-filter 2001:22::/64 orlonger
set policy-options policy-statement adv_global term v6 from route-filter 2001:13:1:1::1/126
orlonger
set policy-options policy-statement adv_global term v6 then next-hop 3021::1
set policy-options policy-statement adv_global term v6 then accept
set policy-options policy-statement adv_vpn1 term v4 from route-filter 32.0.0.0/8 orlonger
set policy-options policy-statement adv_vpn1 term v4 from route-filter 13.1.1.5/30 orlonger
set policy-options policy-statement adv_vpn1 term v4 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v4 then next-hop 3021::1
set policy-options policy-statement adv_vpn1 term v4 then accept
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:32::/64 orlonger
```

```
set policy-options policy-statement adv_vpn1 term v6 from route-filter 2001:13:1:1::5/126
orlonger
set policy-options policy-statement adv_vpn1 term v6 then community set vpn1-target
set policy-options policy-statement adv_vpn1 term v6 then next-hop 3021::1
set policy-options policy-statement adv_vpn1 term v6 then accept
set policy-options policy-statement pplb then load-balance per-packet
set policy-options community vpn1-target members target:100:1
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 13.1.1.5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1022
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 13.1.1.6
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address 2001:13:1:1::5
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1022
set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor 2001:13:1:1::6
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
3021::4
set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
3021::5
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-0/0/2.1
set routing-instances vpn1 route-distinguisher 100:1
set routing-instances vpn1 vrf-export adv_vpn1
set routing-instances vpn1 vrf-target target:100:1
set routing-options source-packet-routing srv6 locator loc1 3021::/64
set routing-options source-packet-routing srv6 no-reduced-srh
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 next-hop 3001::1
set routing-options rib inet6.3 static route abcd::128:53:38:52/128 resolve
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 next-hop 3001::1
set routing-options rib inet6.0 static route abcd::128:53:38:52/128 resolve
set routing-options autonomous-system 100
set routing-options forwarding-table export pplb
set protocols bgp group to-RR type internal
set protocols bgp group to-RR local-address abcd::128:53:35:35
set protocols bgp group to-RR family inet unicast extended-nexthop
set protocols bgp group to-RR family inet unicast advertise-srv6-service
set protocols bgp group to-RR family inet unicast accept-srv6-service
set protocols bgp group to-RR family inet-vpn unicast extended-nexthop
set protocols bgp group to-RR family inet-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet-vpn unicast accept-srv6-service
```

```
set protocols bgp group to-RR family inet6 unicast advertise-srv6-service
set protocols bgp group to-RR family inet6 unicast accept-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to-RR family inet6-vpn unicast accept-srv6-service
set protocols bgp group to-RR export adv_global
set protocols bgp group to-RR neighbor abcd::128:53:38:52
set protocols bgp group to-TG-global-v4 type external
set protocols bgp group to-TG-global-v4 local-address 13.1.1.1
set protocols bgp group to-TG-global-v4 family inet unicast
set protocols bgp group to-TG-global-v4 family inet6 unicast
set protocols bgp group to-TG-global-v4 peer-as 1021
set protocols bgp group to-TG-global-v4 neighbor 13.1.1.2
set protocols bgp group to-TG-global-v6 type external
set protocols bgp group to-TG-global-v6 local-address 2001:13:1:1::1
set protocols bgp group to-TG-global-v6 family inet6 unicast
set protocols bgp group to-TG-global-v6 peer-as 1021
set protocols bgp group to-TG-global-v6 neighbor 2001:13:1:1::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3021::2
set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3021::3
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis source-packet-routing srv6 locator loc1 end-sid 3021::1 flavor usd
set protocols isis level 1 disable
```

Router R3

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.2/30
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:2/64
set interfaces xe-0/0/1:0 unit 0 family inet address 2.5.1.2/30
set interfaces xe-0/0/1:0 unit 0 family iso
set interfaces xe-0/0/1:0 unit 0 family inet6 address 2001:db8::52:0:2/64
set interfaces xe-0/0/1:1 unit 0 family inet address 3.5.1.2/30
set interfaces xe-0/0/1:1 unit 0 family iso
set interfaces xe-0/0/1:1 unit 0 family inet6 address 2001:db8::3:5:2/64
set interfaces xe-0/0/1:2 unit 0 family inet address 4.6.1.1/30
set interfaces xe-0/0/1:2 unit 0 family iso
set interfaces xe-0/0/1:2 unit 0 family inet6 address 2001:db8::4:6:1/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::3/128
set routing-options autonomous-system 100
set protocols isis interface all
```

```
set protocols isis interface fxp0.0 disable
set protocols isis level 1 disable
```

Router R4

```
set chassis network-services enhanced-ip
set interfaces xe-0/0/0:0 unit 0 family inet address 1.6.1.2/30
set interfaces xe-0/0/0:0 unit 0 family iso
set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::6:2/64
set interfaces xe-0/0/0:1 unit 0 family inet address 2.6.1.2/30
set interfaces xe-0/0/0:1 unit 0 family iso
set interfaces xe-0/0/0:1 unit 0 family inet6 address 2001:db8::26:2/64
set interfaces xe-0/0/0:2 unit 0 family inet address 3.6.1.2/30
set interfaces xe-0/0/0:2 unit 0 family iso
set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::3:6:2/64
set interfaces xe-0/0/0:3 unit 0 family inet address 4.6.1.2/30
set interfaces xe-0/0/0:3 unit 0 family iso
set interfaces xe-0/0/0:3 unit 0 family inet6 address 2001:db8::4:6:2/64
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::4/128
set routing-options autonomous-system 100
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis level 1 disable
```

Configure Router R0

Step-by-Step Procedure

To configure SRv6 network programming with Layer 3 VPN services, perform the following steps on Router R0:

1. Configure the device interfaces to enable IP transport.

```
[edit]
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet address 1.4.1.1/30
user@R0# set interfaces xe-0/0/0:0 unit 0 family iso
user@R0# set interfaces xe-0/0/0:0 unit 0 family inet6 address 2001:db8::4:1/64

user@R0# set interfaces xe-0/0/0:2 unit 0 family inet address 1.6.1.1/30
```

```
user@R0# set interfaces xe-0/0/0:2 unit 0 family iso
user@R0# set interfaces xe-0/0/0:2 unit 0 family inet6 address 2001:db8::6:1/64
```

2. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R0# set routing-options router-id 128.53.38.52
user@R0# set routing-options autonomous-system 100
```

3. Enable SRv6 globally and the locator address to indicate the SRv6 capability of the router. SRv6 SID is an IPv6 address that consists of the locator and a function. The routing protocols advertise the locator addresses.

```
[edit]
user@R0# set routing-options source-packet-routing srv6 locator loc1 3001::/64
user@R0# set routing-options source-packet-routing srv6 no-reduced-srh
```

4. Configure an external routing instance VPN1 for both IPv4 and IPv6 traffic. Configure the BGP protocol for VPN1 to enable peering and traffic transport between the provider edge devices.

```
[edit]
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 type external
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 local-address 11.1.1.5
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 family inet6 unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 peer-as 1002
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v4 neighbor 11.1.1.6
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 type external
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 local-address
2001:11:1:1::5
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 family inet6 unicast
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 peer-as 1002
user@R0# set routing-instances vpn1 protocols bgp group to-TG-vpn1-v6 neighbor
2001:11:1:1::6
```

5. Configure the VPN type and a unique route distinguisher for each PE router participating in the routing instance.

```
[edit]
user@R0# set routing-instances vpn1 instance-type vrf
user@R0# set routing-instances vpn1 interface xe-0/0/0:3.1
user@R0# set routing-instances vpn1 route-distinguisher 100:1
user@R0# set routing-instances vpn1 vrf-target target:100:1
```

6. Configure the end-dt4 and end-dt6 SID values for enabling the Layer 3 VPN services.

```
[edit]
user@R0# set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1
end-dt4-sid 3001::4
user@R0# set routing-instances vpn1 protocols bgp source-packet-routing srv6 locator loc1
end-dt6-sid 3001::5
```

7. Define a policy to load-balance packets.

```
[edit]
user@R0# set policy-options policy-statement pplb then load-balance per-packet
user@R0# set policy-options community vpn1-target members target:100:1
user@R0# set policy-options community vpn2-target members target:100:2
```

8. Apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R0# set routing-options forwarding-table export pplb
```

9. Define a policy adv_global to accept routes advertised from R1.

```
[edit]
user@R0# set policy-options policy-statement adv_global term v4 from route-filter
20.0.0.0/8 orlonger
user@R0# set policy-options policy-statement adv_global term v4 then next-hop self
user@R0# set policy-options policy-statement adv_global term v4 then accept
user@R0# set policy-options policy-statement adv_global term v6 from route-filter
2001:20::/64 orlonger
```

```
user@R0# set policy-options policy-statement adv_global term v6 then next-hop self
user@R0# set policy-options policy-statement adv_global term v6 then accept
```

10. Configure BGP on the core-facing interface to establish internal and external peering sessions.

```
[edit]
user@R0# set protocols bgp group to-PE-all type internal
user@R0# set protocols bgp group to-PE-all local-address abcd::128:53:38:52
user@R0# set protocols bgp group to-PE-all family inet unicast extended-nexthop
user@R0# set protocols bgp group to-PE-all family inet unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast extended-nexthop
user@R0# set protocols bgp group to-PE-all export adv_global
user@R0# set protocols bgp group to-PE-all cluster 128.53.38.52
user@R0# set protocols bgp group to-PE-all neighbor abcd::128:53:35:39
user@R0# set protocols bgp group to-PE-all neighbor abcd::128:53:35:35
user@R0# set protocols bgp group to-TG-global-v4 type external
user@R0# set protocols bgp group to-TG-global-v4 local-address 11.1.1.1
user@R0# set protocols bgp group to-TG-global-v4 family inet unicast
user@R0# set protocols bgp group to-TG-global-v4 family inet6 unicast
user@R0# set protocols bgp group to-TG-global-v4
user@R0# set protocols bgp group to-TG-global-v4 neighbor 11.1.1.2
user@R0# set protocols bgp group to-TG-global-v6 type external
user@R0# set protocols bgp group to-TG-global-v6 local-address 2001:11:1:1::1
user@R0# set protocols bgp group to-TG-global-v6 family inet6 unicast
user@R0# set protocols bgp group to-TG-global-v6 peer-as 1001
user@R0# set protocols bgp group to-TG-global-v6 neighbor 2001:11:1:1::2
```

11. Enable the device to advertise the SRv6 services to BGP peers and to accept the routes advertised by the egress provider edge (PE) devices.

```
[edit]
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet-vpn unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6 unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6 unicast accept-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6-vpn unicast advertise-srv6-service
user@R0# set protocols bgp group to-PE-all family inet6-vpn unicast accept-srv6-service
```

12. Enable IS-IS as the interior gateway protocol (IGP) for routing traffic between the core provider routers.

```
[edit]
user@R0# set protocols isis interface all
user@R0# set protocols isis interface fxp0.0 disable
user@R0# set protocols isis source-packet-routing srv6 locator loc1 end-sid 3001::1 flavor
usd
user@R0# set protocols isis level 1 disable
```

13. Configure the end-dt4 and end-dt6 SID value for the prefix segments. End-dt4 is the endpoint SID with decapsulation and IPv4 table lookup and end-dt6 is the endpoint with decapsulation and IPv6 table lookup. BGP allocates these for IPv4 and IPv6 Layer3 VPN services SIDs.

```
[edit]
user@R0# set protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid 3001::2
user@R0# set protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid 3001::3
```

Results

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

```
[edit]
user@R0# show interfaces
xe-0/0/0:0 {
    unit 0 {
        family inet {
            address 1.4.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:db8::4:1/64;
        }
    }
}
xe-0/0/0:1 {
    unit 0 {
        family inet {
```

```
            address 1.5.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:1:4:2::1/126;
        }
    }
}
xe-0/0/0:2 {
    unit 0 {
        family inet {
            address 1.6.1.1/30;
        }
        family iso;
        family inet6 {
            address 2001:db8::6:1/64;
        }
    }
}
```

```
[edit]
user@R0# show protocols
bgp {
    group to-PE-all {
        type internal;
        local-address abcd::128:53:38:52;
        family inet {
            unicast {
                extended-nexthop;
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet-vpn {
            unicast {
                extended-nexthop;
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet6 {
```

```
            unicast {
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        family inet6-vpn {
            unicast {
                advertise-srv6-service;
                accept-srv6-service;
            }
        }
        export adv_global;
        cluster 128.53.38.52;
        neighbor abcd::128:53:35:39;
        neighbor abcd::128:53:35:35;
    }
    group to-TG-global-v4 {
        type external;
        local-address 11.1.1.1;
        family inet {
            unicast;
        }
        family inet6 {
            unicast;
        }
        peer-as 1001;
        neighbor 11.1.1.2;
    }
    group to-TG-global-v6 {
        type external;
        local-address 2001:11:1:1::1;
        family inet6 {
            unicast;
        }
        peer-as 1001;
        neighbor 2001:11:1:1::2;
    }
    source-packet-routing {
        srv6 {
            locator loc1 {
                end-dt4-sid 3001::2;
                end-dt6-sid 3001::3;
            }
```

```
        }
    }
}
isis {
    interface all;
    interface fxp0.0 {
        disable;
    }
    source-packet-routing {
        srv6 {
            locator loc1 {
                end-sid 3001::1 {
                    flavor {
                        usd;
                    }
                }
            }
        }
    }
    level 1 disable;
}
```

```
[edit]
user@R0# show policy-options
policy-options {
    policy-statement adv_global {
        term v4 {
            from {
                route-filter 20.0.0.0/8 orlonger;
            }
            then {
                next-hop self;
                accept;
            }
        }
        term v6 {
            from {
                route-filter 2001:20::/64 orlonger;
            }
            then {
                next-hop self;
```

```
                accept;
            }
        }
    }
    policy-statement pplb {
        then {
            load-balance per-packet;
        }
    }
    community vpn1-target members target:100:1;
    community vpn2-target members target:100:2;
}
```

```
[edit]
user@R0# show routing-options
routing-options {
    source-packet-routing {
        srv6 {
            locator loc1 3001::/64;
            no-reduced-srh;
        }
    }

    router-id 128.53.38.52;
    autonomous-system 100;
    forwarding-table {
        export pplb;
    }
}
```

```
[edit]
user@R0# show routing-instances
routing-instances {
    vpn1 {
        protocols {
            bgp {
                group to-TG-vpn1-v4 {
                    type external;
                    local-address 11.1.1.5;
                    family inet {
```

```
                    unicast;
                }
                family inet6 {
                    unicast;
                }
                peer-as 1002;
                neighbor 11.1.1.6;
            }
            group to-TG-vpn1-v6 {
                type external;
                local-address 2001:11:1:1::5;
                family inet6 {
                    unicast;
                }
                peer-as 1002;
                neighbor 2001:11:1:1::6;
            }
            source-packet-routing {
                srv6 {
                    locator loc1 {
                        end-dt4-sid 3001::4;
                        end-dt6-sid 3001::5;
                    }
                }
            }
        }
        instance-type vrf;
        interface xe-0/0/0:3.1;
        route-distinguisher 100:1;
        vrf-target target:100:1;
    }
}
```

When done configuring the device, enter `commit` from the configuration mode.

Verification

IN THIS SECTION

- Verify that the advertised IPv4 route is installed in the IPv4 table | **943**

- Verify that SRv6 SID is installed in the IPv4 Table | **943**
- Verify that the IPv6 VPN route is installed in the VPN table | **946**
- Verify that the IPv4 VPN route is installed in the VPN table | **947**

Confirm that the configuration is working properly.

Verify that the advertised IPv4 route is installed in the IPv4 table

Purpose

Verify that ingress router R0 has learned the route to the IPv4 prefix 20.0.0.0 from the egress router R1.

Action

From operational mode, run the **show route 20.0.0.0** command on router R0.

```
user@R0> show route 20.0.0.0
inet.0: 59 destinations, 59 routes (59 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.0.0.0/30        *[BGP/170] 09:15:25, localpref 100, from abcd::128:53:37:72
                      AS path: {65501} I, validation-state: unverified
                    >  to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3001::1
                       to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3001::1
                       to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3001::1
```

Meaning

The output confirms that the IPv4 prefix 20.0.0.0 is installed in the inet.0 table.

Verify that SRv6 SID is installed in the IPv4 Table

Purpose

Verify that ingress Router R0 has received and accepted the SRv6 end-dt4 SID 3001::2 from the egress Router R1.

Action

From operational mode, run the **show route 20.0.0.0 extensive** command on Router R0.

```
user@> show route 20.0.0.0 extensive
inet.0: 59 destinations, 59 routes (59 active, 0 holddown, 0 hidden)
20.0.0.0/30 (1 entry, 1 announced)
TSI:
KRT in-kernel 20.0.0.0/30 -> {composite(716)}
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xc5aa39c
                Next-hop reference count: 20
                Source: abcd::128:53:37:72
                Next hop type: List, Next hop index: 1048574
        Next hop: ELNH Address 0xc5a9e88, selected
            Next hop type: Chain, Next hop index: 725
            Address: 0xc5a9e88
            Next-hop reference count: 1
            Next hop: ELNH Address 0xc5a9aa0
            SRV6-Tunnel: Reduced-SRH Encap-mode
             Src: abcd::128:53:35:39 Dest: 3001::1
             Segment-list[0] 3001::1
                Next hop type: Router, Next hop index: 700
                Address: 0xc5a9aa0
                Next-hop reference count: 4
                Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
        Next hop: ELNH Address 0xc5a9eec
            Next hop type: Chain, Next hop index: 726
            Address: 0xc5a9eec
            Next-hop reference count: 1
            Next hop: ELNH Address 0xc5a9c30
            SRV6-Tunnel: Reduced-SRH Encap-mode
             Src: abcd::128:53:35:39 Dest: 3001::1
             Segment-list[0] 3001::1
                Next hop type: Router, Next hop index: 702
                Address: 0xc5a9c30
                Next-hop reference count: 4
                Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
        Next hop: ELNH Address 0xc5aa0e0
            Next hop type: Chain, Next hop index: 727
            Address: 0xc5aa0e0
```

```
Next-hop reference count: 1
Next hop: ELNH Address 0xc5a9780
SRV6-Tunnel: Reduced-SRH Encap-mode
 Src: abcd::128:53:35:39 Dest: 3001::1
 Segment-list[0] 3001::1
    Next hop type: Router, Next hop index: 647
    Address: 0xc5a9780
    Next-hop reference count: 20
    Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
    Protocol next hop: abcd::128:53:37:72
    Composite next hop: 0xbd4e7d0 716 INH Session ID: 0x151
    Indirect next hop: 0xc762204 1048582 INH Session ID: 0x151
    State: <Active int Ext>
    Local AS:   100 Peer AS:   100
    Age: 9:13:44    Metric2: 20
    Validation State: unverified
    ORR Generation-ID: 0
    Task: BGP_100.abcd::128:53:37:72
    Announcement bits (1): 0-KRT
    AS path: {65501}
    Accepted
    SRv6 SID: 3001::2
    Localpref: 100
    Router ID: 128.53.37.72
    Composite next hops: 1
             Protocol next hop: abcd::128:53:37:72 Metric: 20
             Composite next hop: 0xbd4e7d0 716 INH Session ID: 0x151
             Indirect next hop: 0xc762204 1048582 INH Session ID: 0x151
             Indirect path forwarding next hops: 3
                    Next hop type: List
                    Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
                    Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
                    Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
                    abcd::128:53:37:72/128 Originating RIB: inet6.3
                      Metric: 20 Node path count: 1
                      Indirect next hops: 1
                    Protocol next hop: 3001::1 Metric: 20
                    Inode flags: 0x206 path flags: 0x0
                    Path fnh link: 0xc3bf4c0 path inh link: 0x0
                    Indirect next hop: 0xc76cd04 - INH Session ID: 0x0
                    Indirect path forwarding next hops: 3
                            Next hop type: List
                            Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
```

```
                                        Next hop: fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0
                                        Next hop: fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0
                                        3001::1/128 Originating RIB: inet6.3
                                          Metric: 20 Node path count: 1
                                          Forwarding nexthops: 3
                                                Next hop type: List
                                                Next hop: fe80::2e6b:f5ff:fe28:2bcb via ae0.0
                                                Next hop: fe80::2e6b:f5ff:fe28:2b04 via
xe-0/0/0:2.0
                                                Next hop: fe80::2e6b:f5ff:fe73:1e01 via
xe-0/0/0:3.0
```

Meaning

The output displays the SRv6 SID and confirms that an SRv6 tunnel is established between Routers R0 and R1.

Verify that the IPv6 VPN route is installed in the VPN table

Purpose

Verify that ingress router R0 has learned the route to the VPN IPv6 prefix 2001::30::/126 from the egress router R1.

Action

From operational mode, run the **show route 2001:30::** command on router R0.

```
user@R0> show route 2001:30::
vpn1.inet6.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:30::/126      *[BGP/170] 09:15:40, localpref 100, from abcd::128:53:37:72
                      AS path: {65502} I, validation-state: unverified
                    >  to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3001::1
                       to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3001::1
                       to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3001::1
```

Meaning

The output confirms that the route details for the prefix 2001:30::/126 are installed in the vpn.inet6.0 table.

Verify that the IPv4 VPN route is installed in the VPN table

Purpose

Verify that ingress router R0 has learned the route to the VPN IPv4 prefix 30.0.0.0 from the egress router R1.

Action

From operational mode, run the **show route 30.0.0.0** command on router R0.

```
user@R0> show route 30.0.0.0
vpn1.inet.0: 34 destinations, 34 routes (34 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

30.0.0.0/30        *[BGP/170] 09:15:29, localpref 100, from abcd::128:53:37:72
                      AS path: {65502} I, validation-state: unverified
                       to fe80::2e6b:f5ff:fe28:2bcb via ae0.0, SRV6-Tunnel, Dest: 3001::1
                       to fe80::2e6b:f5ff:fe28:2b04 via xe-0/0/0:2.0, SRV6-Tunnel, Dest: 3001::1
                     >  to fe80::2e6b:f5ff:fe73:1e01 via xe-0/0/0:3.0, SRV6-Tunnel, Dest: 3001::1
```

Meaning

The output confirms that the IPv4 prefix 30.0.0.0 is installed in the vpn.inet.0 table.

SEE ALSO

srv6

advertise-srv6-service

accept-srv6-service

Understanding SR-TE Policy for SRv6 Tunnel

IN THIS SECTION

- Benefits of SRv6 TE Policy | **948**
- SRv6 TE Policy Overview | **948**
- What is a Segment Routing Extension Header (SRH)? | **949**
- TI-LFA for SRv6 TE | **951**
- Layer 3 VPN Services over the SRv6 Core | **951**
- Advertising Layer 3 VPN Services to BGP Peers | **952**
- Supported and Unsupported Features for SRv6 Network Programming in SR-TE | **952**

Benefits of SRv6 TE Policy

- SRv6 TE provides flexibility to leverage segment routing without deploying MPLS. Such networks depend only on the IPv6 headers and header extensions for transmitting data. This is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS.
- Ensures a seamless deployment without any major hardware or software upgrade in a core IPv6 network, thereby enhancing scalability.
- Utilizes IS-IS SRv6 SIDs to form the segment lists. Therefore, it leverages the TI-LFA paths of IS-IS SRv6 SIDs and can form backup paths based on the IGP.
- Leverages IS-IS weighted equal cost multipath (ECMP) and can also have its own ECMPs on individual segment lists to form hierarchical weighted ECMPs that performs load balancing at a granular level.

SRv6 TE Policy Overview

An SR-TE policy contains one or more SR-TE tunnels either configured statically or contributed by different tunnel sources namely PCEP, BGP-SRTE, DTM. Starting in Junos OS Release 21.3R1, Junos OS supports SRv6 data plane with statically configured SR-TE policy.

In an SRv6 TE policy:

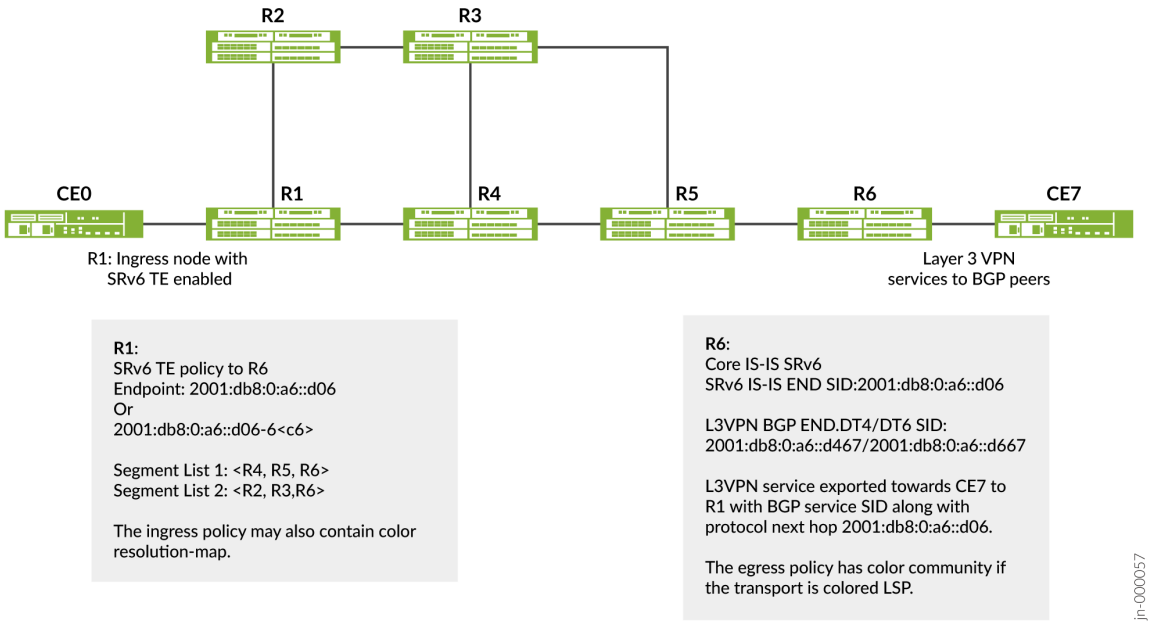
- IS-IS configuration populates the core.
- SRv6 TE tunnel configuration populates the transport.

- BGP network layer reachability information (NLRI) populates the service.

After the creation of the SRv6 TE data plane, you can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the data plane. The desired payload can be of IPv4 or IPv6.

Figure 61 on page 949 depicts an SRv6 TE topology in which R1 is the ingress node with SRv6 TE policy configured to R6. R6 is the egress node with Layer 3 VPN services to BGP peers configured. The core constitutes IS-IS SRv6. The egress Router R6 advertises the L3VPN SID to ingress Router R1, which accepts and updates the VRF table. R6 is configured with 2001:db8:0:a6::d06 as end-sid and the L3VPN service is exported towards CE7 to R1 with 2001:db8:0:a6::d06 as next hop. There are two segment lists: <R4, R5, R6> and <R2, R3, R6>.

Figure 61: SRv6 TE Sample Topology



What is a Segment Routing Extension Header (SRH)?

A Segment Identifier represents a specific segment in a segment routing domain. In an IPv6 network, the SID-type used is a 128-bit IPv6 address also referred to as an SRv6 Segment or SRv6 SID. SRv6 stacks up these IPv6 addresses instead of MPLS labels in a segment routing extension header. The Segment Routing Extension Header (SRH) is a type of IPv6 routing extension header. Typically, the SRH contains a segment list encoded as an SRv6 SID. An SRv6 SID consists of the following parts:

- **Locator**— Locator is the first part of a SID that consists of the most significant bits representing the address of a particular SRv6 node. The locator is very similar to a network address that provides a

route to its parent node. The IS-IS protocol installs the locator route in the `inet6.0` routing table. IS-IS routes the segment to its parent node, which subsequently performs a function defined in the other part of the SRv6 SID. You can also specify the algorithm associated with this locator.

- **Function**—The other part of the SID defines a function that is performed locally on the node that is specified by the locator. There are several functions that have already been defined in the Internet draft draft-ietf-spring-srv6-network-programming-07draft, *SRv6 Network Programming*. However, we have implemented the following functions are available on Junos OS that are signalled in IS-IS. IS-IS installs these function SIDs in the `inet6.0` routing table.

  - **End**— An endpoint function for SRv6 instantiation of a Prefix SID. It does not allow for decapsulation of an outer header for the removal of an SRH. Therefore, an End SID cannot be the last SID of a SID list and cannot be the Destination Address (DA) of a packet without an SRH (unless combined with the PSP, USP or USD flavors).

  - **End.X**— An endpoint X function is an SRv6 instantiation of an adjacent SID. It is a variant of the endpoint function with Layer 3 cross-connect to an array of Layer 3 adjacencies.

  You can specify End SID behavior such as Penultimate Segment Pop (PSP), Ultimate Segment Pop (USP) or Ultimate Segment Decapsulation (USD).

  - **PSP**— When the last SID is written in the destination address, the End and End.X functions with the PSP flavor pop the top-most SRH. Subsequent stacked SRHs may be present but are not processed as part of the function.

  - **USP**— When the next header is an SRH and there are no more segments left, the IS-IS protocol pops the top SRH, looks up the updated destination address and forwards the packet based on match table entry.

  - **USD**— When the next Header in the packet is 41 or is an SRH and there are no more segments left, then IS-IS pops the outer IPv6 header and its extension headers, looks up the exposed inner IP destination address and forwards the packet to the matched table entry.

For example, you can have an SRv6 SID where 2001::19:db8:AC05:FF01:FF01: is the locator and A000:B000:C000:A000 is the function:

Table 7: 128-bit SRv6 SID

Locator	Function
2001::db8:19:AC05:FF01:FF01	A000:B000:C000:A000

TI-LFA for SRv6 TE

Topology Independent- Loop Free Alternate (TI-LFA) establishes a Fast Reroute (FRR) path that is aligned to a post-convergence path. An SRv6-capable node inserts a single segment into the IPv6 header or multiple segments into the SRH. Multiple SRHs can significantly raise the encapsulation overhead, which can sometimes be more than the actual packet payload. Therefore, by default, Junos OS supports SRv6 TE tunnel encapsulation with reduced SRH. The point-of-local repair (PLR) adds the FRR path information to the SRH containing the SRv6 SIDs.

The TI-LFA backup path is represented as a group of SRv6 SIDs inside an SRH. At the ingress router, IS-IS encapsulates the SRH in a fresh IPv6 header. However, at transit routers, IS-IS inserts the SRH into the data traffic in the following manner:

- **Encap Mode**— In the encap mode, the original IPv6 packet is encapsulated and transported as the inner packet of an IPv6-in-IPv6 encapsulated packet. The outer IPv6 packet carries the SRH with the segment list. The original IPv6 packet travels unmodified in the network. By default, Junos OS supports SRv6 tunnel encapsulation in reduced SRH. However, you can choose one of the following tunnel encapsulation methods:
  - **Reduced SRH (default)**— With the reduced SRH mode, ifbecause there is only one SID, there is no SRH added and the last SID is copied into the IPV6 destination address. You cannot preserve the entire SID list in the SRH with a reduced SRH.
  - **Non-reduced SRH**— You can configure the non-reduced SRH tunnel encapsulation mode when you and might still want to preserve the entire SID list in the SRH.

Because the core network of statically configured SRv6 TE LSP is formed by IS-IS SRv6, the IS-IS SRv6 TILFA can be leveraged using SRv6 TE segments.

Layer 3 VPN Services over the SRv6 Core

When connecting to the egress PE, the ingress PE encapsulates the payload in an outer IPv6 header where the destination address is the SRv6 service SID associated with the related BGP route update. The egress PE sets the next hop to one of its IPv6 addresses that is also the SRv6 locator from which the SRv6 service SID is allocated. Multiple routes can resolve through the same segment routing policy.

Figure 62: SRv6 Packet Encapsulation

Starting in Junos OS Release 20.4R1, you can configure BGP-based Layer 3 service over the SRv6 core. You can enable Layer 3 overlay services with BGP as the control plane and SRv6 as the dataplane.

Advertising Layer 3 VPN Services to BGP Peers

BGP advertises the reachability of prefixes of a particular service from an egress PE device to ingress PE nodes. BGP messages exchanged between PE devices carry SRv6 service SIDs, which BGP uses to interconnect PE devices to form VPN sessions. For Layer 3 VPN services where BGP uses a per-VRF SID allocation, the same SID is shared across multiple network layer reachability information (NLRI) address families.

Egress PE devices that support SRv6-based Layer 3 services advertise overlay service prefixes along with a service SID. The BGP ingress node receives these advertisements and adds the prefix to the corresponding virtual routing and forwarding (VRF) table.

Supported and Unsupported Features for SRv6 Network Programming in SR-TE

SRv6 TE currently supports::

- IPv4 and IPv6 payloads.
- Upto 6 SIDs in reduced mode at the ingress router and upto 5 SIDs in non-reduced mode at the ingress.
- Encapsulation mode on the ingress router.
- `preserve-nexthop-hierarchy` configuration under resolver for platform layer to be able to combine SIDs from SR-TE and IGP routes.

SRv6 TE currently does not support::

- Local CSPF capabilities for SRv6 policies.
- IPv4-colored tunnel endpoint.
- sBFD and Telemetry.
- PCE initiated and delegated SRv6 LSPs.
- Auto-translation with SRv6 SIDs.
- LDP tunneling with an SRv6 policy.
- Logical Systems.
- SR-TE binding SID for an SR-TE tunnel.

- Ping or OAM for SRTE SRv6.
- Any Static IPv4 route over SRv6 TE tunnel.
- Insert mode for SRv6 TE.
- SRv6 flexible algorithm for SRv6 TE LSPs.

SEE ALSO

> *source-packet-routing (Segment Routing Traffic Engineering)*
>
> *preserve-nexthop-hierarchy (SRv6 TE)*
>
> *segment-list*

Example: Configuring Static SR-TE Policy for an SRv6 Tunnel

IN THIS SECTION

- Overview | **953**
- Requirements | **954**
- Configuration | **955**
- Verification | **979**

Overview

IN THIS SECTION

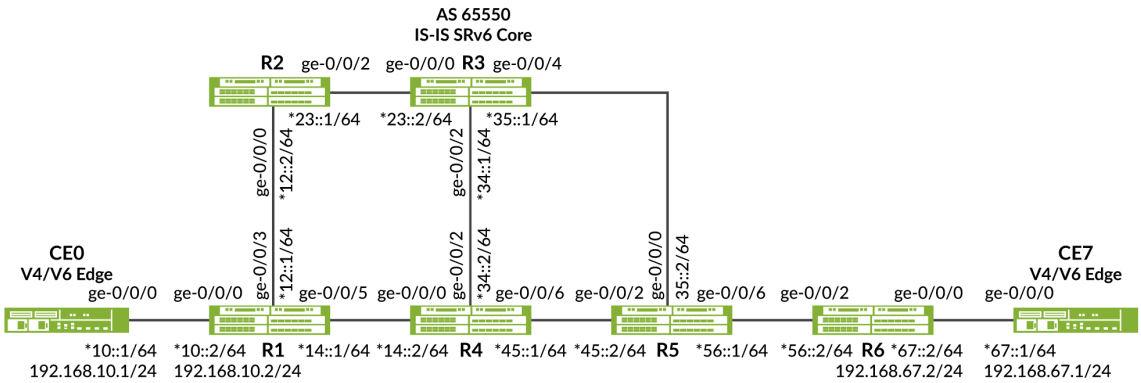
- Topology | **954**

This example shows how to configure static SR-TE policy for an SRv6 Tunnel. This SRv6 TE policy is useful for service providers whose networks are predominantly IPv6 and have not deployed MPLS. Such networks depend only on the IPv6 headers and header extensions for transmitting data. SRv6 network programming provides flexibility to leverage segment routing without deploying MPLS.

Topology

The following illustration depicts an SRv6 TE topology in which the device R1 and device R6 are the ingress and egress routers that support IPv4 or IPv6 devices CE1 and CE2. The devices R2, R3, R4, and R5 comprise an IPv6 only provider core network. All the devices belong to the same autonomous system. IS-IS is the interior gateway protocol in the IPv6 core and is configured to support SRv6. In this example the egress device R6 advertises the L3VPN SID to the ingress device R1, which accepts and updates the VRF table. The device R6 is configured with 2001:db8:0:a6::d06 as end-sid and the L3VPN service is exported towards CE7 to R1 with 2001:db8:0:a6::d06 as next hop. There are two segment lists: <R4, R5, R6> and <R2, R3, R6>.

Figure 63: SRv6 TE Topology



Router	Loopback Address
R1	2001:db8:1:255::1/128
R2	2001:db8:2:255::2/128
R3	2001:db8:3:255::3/128
R4	2001:db8:3:255::4/128
R5	2001:db8:3:255::5/128
R6	2001:db8:3:255::6/128
CE0	10.100.10.1/32
CE7	10.100.10.7/32

* IPv6 Address Prefix: 2001:db8:

jn-000107

Requirements

This example uses the following hardware and software components:

- Six MX Series routers.
- Junos OS Release 21.3R1 or later.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 955
- Configuring Device R1 | 965
- Results | 972

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 , and then enter commit from configuration mode.

Device R1

```
set interfaces ge-0/0/0 unit 0 description R1_to_CE0
set interfaces ge-0/0/0 unit 0 family inet address 192.168.10.2/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10::2/64
set interfaces ge-0/0/3 unit 0 description R1_to_R2
set interfaces ge-0/0/3 unit 0 family iso
set interfaces ge-0/0/3 unit 0 family inet6 address 2001:db8:12::1/64
set interfaces ge-0/0/3 unit 0 description R1_to_R4
set interfaces ge-0/0/5 unit 0 family iso
set interfaces ge-0/0/5 unit 0 family inet6 address 2001:db8:14::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0001.0101.0100
set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::1/128
set policy-options policy-statement to_CE0_community_export term 0 then community add
to_CE0_community
set policy-options policy-statement to_CE0_community_export term 0 then next-hop
2001:db8:0:a1::d01
set policy-options policy-statement to_CE0_community_export term 0 then accept
set policy-options policy-statement to_CE0_community_import term 0 from community
to_CE0_community
set policy-options policy-statement to_CE0_community_import term 0 then accept
set policy-options policy-statement v4vpn1_res_map1 term 1 from protocol bgp
set policy-options policy-statement v4vpn1_res_map1 term 1 then accept
```

```
set policy-options policy-statement v4vpn1_res_map1 term 1 then resolution-map map1
set policy-options policy-statement v6vpn1_res_map1 term 1 from family inet6-vpn
set policy-options policy-statement v6vpn1_res_map1 term 1 from protocol bgp
set policy-options policy-statement v6vpn1_res_map1 term 1 then accept
set policy-options policy-statement v6vpn1_res_map1 term 1 then resolution-map map1
set policy-options policy-statement LBPP term 1 then load-balance per-packet
set policy-options policy-statement mpath-resolve then multipath-resolve
set policy-options resolution-map map1 mode ip-color
set policy-options community to_CE0_community members target:65500:1
set routing-instances to_CE0 instance-type vrf
set routing-instances to_CE0 protocols bgp group to_CE0_v6 type external
set routing-instances to_CE0 protocols bgp group to_CE0_v6 as-override
set routing-instances to_CE0 protocols bgp group to_CE0_v6 peer-as 65000
set routing-instances to_CE0 protocols bgp group to_CE0_v6 neighbor 2001:db8:10::1
set routing-instances to_CE0 protocols bgp group to_CE0_v4 type external
set routing-instances to_CE0 protocols bgp group to_CE0_v4 as-override
set routing-instances to_CE0 protocols bgp group to_CE0_v4 peer-as 65000
set routing-instances to_CE0 protocols bgp group to_CE0_v4 neighbor 192.168.10.1
set routing-instances to_CE0 protocols bgp source-packet-routing srv6 locator loc1 end-dt4-sid
2001:db8:0:a1::d410
set routing-instances to_CE0 protocols bgp source-packet-routing srv6 locator loc1 end-dt6-sid
2001:db8:0:a1::d610
set routing-instances to_CE0 interface ge-0/0/0.0
set routing-instances to_CE0 route-distinguisher 192.168.255.11:1
set routing-instances to_CE0 vrf-import to_CE0_community_import
set routing-instances to_CE0 vrf-export to_CE0_community_export
set routing-options source-packet-routing srv6 locator loc1 2001:db8:0:a1::/112
set routing-options resolution preserve-nexthop-hierarchy
set routing-options resolution rib bgp.l3vpn-inet6.0 resolution-ribs inet6.3
set routing-options resolution rib bgp.l3vpn-inet6.0 inet6-resolution-ribs inet6.3
set routing-options resolution rib bgp.l3vpn-inet6.0 import mpath-resolve
set routing-options resolution rib bgp.l3vpn-inet6.0 inet6-import mpath-resolve
set routing-options resolution rib bgp.l3vpn-inet6.0 inet6color-import mpath-resolve
set routing-options resolution rib bgp.l3vpn.0 import mpath-resolve
set routing-options resolution rib bgp.l3vpn.0 inet6-import mpath-resolve
set routing-options resolution rib inet6.0 import mpath-resolve
set routing-options resolution rib inet.0 import mpath-resolve
set routing-options router-id 192.168.255.11
set routing-options autonomous-system 65500
set routing-options forwarding-table srv6-chain-merge
set routing-options forwarding-table export LBPP
set protocols bgp group to_R6_ibgpv6 type internal
set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast accept-srv6-service
```

```
set protocols bgp group to_R6_ibgpv6 family inet6 unicast extended-nexthop-color
set protocols bgp group to_R6_ibgpv6 family inet6-vpn unicast advertise-srv6-service
set protocols bgp group to_R6_ibgpv6 family inet6-vpn unicast accept-srv6-service
set protocols bgp group to_R6_ibgpv6 local-address 2001:db8:1:255::1
set protocols bgp group to_R6_ibgpv6 import v4vpn1_res_map1
set protocols bgp group to_R6_ibgpv6 import v6vpn1_res_map1
set protocols bgp group to_R6_ibgpv6 family inet unicast extended-nexthop
set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast extended-nexthop
set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast advertise-srv6-serviceset protocols
bgp group to_R6_ibgpv6 neighbor 2001:db8:6:255::6
set protocols bgp multipath
set protocols isis interface ge-0/0/3.0 level 2 disable
set protocols isis interface ge-0/0/3.0 level 1 srv6-adjacency-segment unprotected locator loc1
end-x-sid 2001:db8:0:a1::1a12 flavor psp
set protocols isis interface ge-0/0/3.0 level 1 srv6-adjacency-segment unprotected locator loc1
end-x-sid 2001:db8:0:a1::1a12 flavor usd
set protocols isis interface ge-0/0/3.0 point-to-point
set protocols isis interface ge-0/0/5.0 level 2 disable
set protocols isis interface ge-0/0/5.0 level 1 post-convergence-lfa
set protocols isis interface ge-0/0/5.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols isis source-packet-routing node-segment ipv6-index 101
set protocols isis source-packet-routing srv6 locator loc1 end-sid 2001:db8:0:a1::d01 flavor psp
set protocols isis source-packet-routing srv6 locator loc1 end-sid 2001:db8:0:a1::d01 flavor usp
set protocols isis source-packet-routing srv6 locator loc1 end-sid 2001:db8:0:a1::d01 flavor usd
set protocols isis backup-spf-options use-post-convergence-lfa
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols source-packet-routing segment-list end-sids-segment srv6
set protocols source-packet-routing segment-list end-sids-segment hop1 srv6-sid
2001:db8:0:a4::d04
set protocols source-packet-routing segment-list end-sids-segment hop2 srv6-sid
2001:db8:0:a5::d05
set protocols source-packet-routing segment-list end-sids-segment hop3 srv6-sid
2001:db8:0:a6::d06
set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-sid srv6
set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-sid hop1 srv6-
sid 2001:db8:0:a2::1a23
set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-sid hop2 srv6-
sid 2001:db8:0:a3::1a34
```

```
set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-sid hop3 srv6-
sid 2001:db8:0:a6::d06
set protocols source-packet-routing srv6
set protocols source-packet-routing source-routing-path nc_path_R1R6 srv6
set protocols source-packet-routing source-routing-path nc_path_R1R6 to 2001:db8:0:a6::d06
set protocols source-packet-routing source-routing-path nc_path_R1R6 from 2001:db8:1:255::1
set protocols source-packet-routing source-routing-path nc_path_R1R6 primary end-sids-segment
weight 40
set protocols source-packet-routing source-routing-path nc_path_R1R6 primary end-x-sids-segment-
last-sid-end-sid weight 30
set protocols source-packet-routing source-routing-path c_path_R1R6 srv6
set protocols source-packet-routing source-routing-path c_path_R1R6 to 2001:db8:0:a6::d06
set protocols source-packet-routing source-routing-path c_path_R1R6 from 2001:db8:1:255::1
set protocols source-packet-routing source-routing-path c_path_R1R6 color 6
set protocols source-packet-routing source-routing-path c_path_R1R6 primary end-sids-segment
weight 40
set protocols source-packet-routing source-routing-path c_path_R1R6 primary end-x-sids-segment-
last-sid-end-sid weight 30
```

Device R2

```
set interfaces ge-0/0/0 unit 0 description R2_To_R1
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:12::2/64
set interfaces ge-0/0/2 unit 0 description R2_To_R3
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:23::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0002.0202.0200
set interfaces lo0 unit 0 family inet6 address 2001:db8:2:255::2/128
set routing-options source-packet-routing srv6 locator loc2 2001:db8:0:a2::/112
set routing-options router-id 192.168.255.22
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc2
end-x-sid 2001:db8:0:a2::1a12 flavor psp
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc2
end-x-sid 2001:db8:0:a2::1a12 flavor usd
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc2
end-x-sid 2001:db8:0:a2::1a23 flavor psp
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc2
end-x-sid 2001:db8:0:a2::1a23 flavor usd
```

```
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 passive
set protocols isis source-packet-routing node-segment ipv6-index 110
set protocols isis source-packet-routing srv6 locator loc2 end-sid 2001:db8:0:a2::d02 flavor psp
set protocols isis source-packet-routing srv6 locator loc2 end-sid 2001:db8:0:a2::d02 flavor usp
set protocols isis source-packet-routing srv6 locator loc2 end-sid 2001:db8:0:a2::d02 flavor usd
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

Device R3

```
set interfaces ge-0/0/0 unit 0 description R3_To_R2
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:23::2/64
set interfaces ge-0/0/2 unit 0 description R3_To_R4
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:34::1/64
set interfaces ge-0/0/4 unit 0 description R3_To_R5
set interfaces ge-0/0/4 unit 0 family iso
set interfaces ge-0/0/4 unit 0 family inet6 address 2001:db8:35::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0003.0303.0300
set interfaces lo0 unit 0 family inet6 address 2001:db8:3:255::3/128
set routing-options source-packet-routing srv6 locator loc3 2001:db8:0:a3::/112
set routing-options router-id 192.168.255.33
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a23 flavor psp
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a23 flavor usd
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a34 flavor psp
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a34 flavor usd
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/4.0 level 2 disable
set protocols isis interface ge-0/0/4.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a35 flavor psp
```

```
set protocols isis interface ge-0/0/4.0 level 1 srv6-adjacency-segment unprotected locator loc3
end-x-sid 2001:db8:0:a3::1a35 flavor usd
set protocols isis interface ge-0/0/4.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0set protocols isis source-packet-routing node-segment ipv6-
index 120
set protocols isis source-packet-routing srv6 locator loc3 end-sid 2001:db8:0:a3::d03 flavor usp
set protocols isis source-packet-routing srv6 locator loc3 end-sid 2001:db8:0:a3::d03 flavor usd
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

Device R4

```
set interfaces ge-0/0/0 unit 0 description R4_To_R1
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:14::2/64
set interfaces ge-0/0/2 unit 0 description R4_To_R3
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:34::2/64
set interfaces ge-0/0/6 unit 0 description R4_To_R5
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family inet6 address 2001:db8:45::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0004.0404.0400
set interfaces lo0 unit 0 family inet6 address 2001:db8:4:255::4/128
set routing-options source-packet-routing srv6 locator loc4 2001:db8:0:a4::/112
set routing-options router-id 192.168.255.44
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc4
end-x-sid 2001:db8:0:a4::1a41 flavor psp
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc4
end-x-sid 2001:db8:0:a4::1a41 flavor usd
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc4
end-x-sid 2001:db8:0:a4::1a34 flavor psp
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc4
end-x-sid 2001:db8:0:a4::1a34 flavor usd
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface ge-0/0/6.0 level 2 disable
set protocols isis interface ge-0/0/6.0 level 1 srv6-adjacency-segment unprotected locator loc4
```

```
end-x-sid 2001:db8:0:a4::1a45 flavor psp
set protocols isis interface ge-0/0/6.0 level 1 srv6-adjacency-segment unprotected locator loc4
end-x-sid 2001:db8:0:a4::1a45 flavor usd
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0set protocols isis source-packet-routing node-segment ipv6-
index 130
set protocols isis source-packet-routing srv6 locator loc4 end-sid 2001:db8:0:a4::d04 flavor psp
set protocols isis source-packet-routing srv6 locator loc4 end-sid 2001:db8:0:a4::d04 flavor usp
set protocols isis source-packet-routing srv6 locator loc4 end-sid 2001:db8:0:a4::d04 flavor usd
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

Device R5

```
set interfaces ge-0/0/0 unit 0 description R5_To_R3
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:35::2/64
set interfaces ge-0/0/2 unit 0 description R5_To_R4
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:45::2/64
set interfaces ge-0/0/6 unit 0 description R5_To_R6
set interfaces ge-0/0/6 unit 0 family iso
set interfaces ge-0/0/6 unit 0 family inet6 address 2001:db8:56::1/64
set interfaces lo0 unit 0 family iso address 49.0001.0005.0505.0500
set interfaces lo0 unit 0 family inet6 address 2001:db8:5:255::5/128
set routing-options source-packet-routing srv6 locator loc5 2001:db8:0:a5::/112
set routing-options router-id 192.168.255.55
set protocols isis interface ge-0/0/0.0 level 2 disable
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a35 flavor psp
set protocols isis interface ge-0/0/0.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a35 flavor usd
set protocols isis interface ge-0/0/0.0 point-to-point
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a45 flavor psp
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a45 flavor usd
set protocols isis interface ge-0/0/2.0 point-to-point
```

```
set protocols isis interface ge-0/0/6.0 level 2 disable
set protocols isis interface ge-0/0/6.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a56 flavor psp
set protocols isis interface ge-0/0/6.0 level 1 srv6-adjacency-segment unprotected locator loc5
end-x-sid 2001:db8:0:a5::1a56 flavor usd
set protocols isis interface ge-0/0/6.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols isis source-packet-routing node-segment ipv6-index 150
set protocols isis source-packet-routing srv6 locator loc5 end-sid 2001:db8:0:a5::d05 flavor psp
set protocols isis source-packet-routing srv6 locator loc5 end-sid 2001:db8:0:a5::d05 flavor usp
set protocols isis source-packet-routing srv6 locator loc5 end-sid 2001:db8:0:a5::d05 flavor usd
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

Device R6

```
set interfaces ge-0/0/0 unit 0 description R6_To_CE7
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:67::2/64
set interfaces ge-0/0/2 unit 0 description R6_To_R5
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:67::2/64
set interfaces ge-0/0/2 unit 0 description R6_To_R5
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:56::2/64
set interfaces lo0 unit 0 family iso address 49.0001.0006.0606.0600
set interfaces lo0 unit 0 family inet6 address 2001:db8:6:255::6/128
set policy-options policy-statement LBPP term 1 then load-balance per-packet
set policy-options policy-statement mpath-resolve then multipath-resolve
set policy-options policy-statement to_CE7_community_export term 0 then community add
to_CE7_community
set policy-options policy-statement to_CE7_community_export term 0 then community add
to_ce7_color_com
set policy-options policy-statement to_CE7_community_export term 0 then next-hop
2001:db8:0:a6::d06
set policy-options policy-statement to_CE7_community_export term 0 then accept
set policy-options policy-statement to_CE7_community_import term 0 from community
to_CE7_community
```

```
set policy-options policy-statement to_CE7_community_import term 0 then accept
set policy-options community to_CE7_community members target:65500:1
set policy-options community to_ce7_color_com members color:1:6
set routing-instances to_CE7 instance-type vrf
set routing-instances to_CE7 protocols bgp group to_CE7_v6 type external
set routing-instances to_CE7 protocols bgp group to_CE7_v6 as-override
set routing-instances to_CE7 protocols bgp group to_CE7_v6 peer-as 65000
set routing-instances to_CE7 protocols bgp group to_CE7_v6 neighbor 2001:db8:67::1
set routing-instances to_CE7 protocols bgp group to_CE7_v4 type external
set routing-instances to_CE7 protocols bgp group to_CE7_v4 as-override
set routing-instances to_CE7 protocols bgp group to_CE7_v4 peer-as 65000
set routing-instances to_CE7 protocols bgp group to_CE7_v4 neighbor 192.168.67.1
set routing-instances to_CE7 protocols bgp source-packet-routing srv6 locator loc8 end-dt4-sid
2001:db8:0:a6::d467
set routing-instances to_CE7 protocols bgp source-packet-routing srv6 locator loc8 end-dt6-sid
2001:db8:0:a6::d667
set routing-instances to_CE7 interface ge-0/0/0.0
set routing-instances to_CE7 route-distinguisher 192.168.255.66:6
set routing-instances to_CE7 vrf-import to_CE7_community_import
set routing-instances to_CE7 vrf-export to_CE7_community_export
set routing-options source-packet-routing srv6 locator loc8 2001:db8:0:a6::/112
set routing-options resolution preserve-nexthop-hierarchy
set routing-options resolution rib bgp.l3vpn-inet6.0 resolution-ribs inet6.3
set routing-options resolution rib bgp.l3vpn-inet6.0 inet6-resolution-ribs inet6.3
set routing-options resolution rib bgp.l3vpn-inet6.0 import mpath-resolve
set routing-options resolution rib bgp.l3vpn.0 inet6-resolution-ribs inet6.3
set routing-options resolution rib bgp.l3vpn.0 import mpath-resolve
set routing-options resolution rib inet6.0 import mpath-resolve
set routing-options resolution rib inet.0 import mpath-resolve
set routing-options router-id 192.168.255.66
set routing-options autonomous-system 65500
set routing-options forwarding-table srv6-chain-merge
set routing-options forwarding-table export LBPP
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 2001:db8:6:255::6
set protocols bgp group ibgp family inet unicast extended-nexthop
set protocols bgp group ibgp family inet-vpn unicast extended-nexthop
set protocols bgp group ibgp family inet-vpn unicast advertise-srv6-service
set protocols bgp group ibgp family inet-vpn unicast accept-srv6-service
set protocols bgp group ibgp family inet6 unicast extended-nexthop-color
set protocols bgp group ibgp family inet6-vpn unicast advertise-srv6-service
set protocols bgp group ibgp family inet6-vpn unicast accept-srv6-service
set protocols bgp group ibgp neighbor 2001:db8:1:255::1
```

```
set protocols bgp multipath
set protocols isis interface ge-0/0/2.0 level 2 disable
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc8
end-x-sid 2001:db8:0:a6::1a56 flavor psp
set protocols isis interface ge-0/0/2.0 level 1 srv6-adjacency-segment unprotected locator loc8
end-x-sid 2001:db8:0:a6::1a56 flavor usd
set protocols isis interface ge-0/0/2.0 point-to-point
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols isis source-packet-routing srgb start-label 400000
set protocols isis source-packet-routing srgb index-range 5000
set protocols isis source-packet-routing node-segment ipv6-index 170
set protocols isis source-packet-routing srv6 locator loc8 end-sid 2001:db8:0:a6::d06 flavor psp
set protocols isis source-packet-routing srv6 locator loc8 end-sid 2001:db8:0:a6::d06 flavor usp
set protocols isis source-packet-routing srv6 locator loc8 end-sid 2001:db8:0:a6::d06 flavor usd
set protocols isis backup-spf-options use-source-packet-routing
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
```

Device CE0

```
set interfaces ge-0/0/0 unit 0 description CE0_To_R1
set interfaces ge-0/0/0 unit 0 family inet address 192.168.10.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10::1/64
set interfaces lo0 unit 0 family inet address 10.100.10.1/32
set interfaces lo0 unit 0 family iso address 49.0001.000a.0a0a.0a00
set interfaces lo0 unit 0 family inet6 address 2001:db8:10:255::10/128
set policy-options policy-statement BGP_export term 0 from protocol direct
set policy-options policy-statement BGP_export term 0 from interface lo0.0
set policy-options policy-statement BGP_export term 0 then accept
set routing-options rib inet6.0 static route 0::0/0 next-hop 2001:db8:10::2
set routing-options rib inet.0 static route 0.0.0.0/0 next-hop 192.168.10.2
set routing-options router-id 10.100.10.1
set routing-options autonomous-system 65000
set protocols bgp group eBGPv6 type external
set protocols bgp group eBGPv6 export BGP_export
set protocols bgp group eBGPv6 peer-as 65500
set protocols bgp group eBGPv6 neighbor 2001:db8:10::2
set protocols bgp group eBGPv4 type external
set protocols bgp group eBGPv4 export BGP_export
```

```
set protocols bgp group eBGPv4 peer-as 65500
set protocols bgp group eBGPv4 neighbor 192.168.10.2
set protocols isis interface all level 2 disable
set protocols isis interface fxp0.0 disable
```

Device CE7

```
set system host-name CE7
set system services netconf ssh
set system ports console log-out-on-disconnect
set interfaces ge-0/0/0 unit 0 description CE7_To_R6
set interfaces ge-0/0/0 unit 0 family inet address 192.168.67.1/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:67::1/64
set interfaces lo0 unit 0 family inet address 10.100.10.7/32
set interfaces lo0 unit 0 family iso address 49.0001.0007.0707.0700
set interfaces lo0 unit 0 family inet6 address 2001:db8:7:255::7/128
set policy-options policy-statement BGP_export term 0 from protocol direct
set policy-options policy-statement BGP_export term 0 from interface lo0.0
set policy-options policy-statement BGP_export term 0 then accept
set routing-options rib inet6.0 static route 0::0/0 next-hop 2001:db8:67::2
set routing-options rib inet.0 static route 0.0.0.0/0 next-hop 192.168.67.2
set routing-options router-id 10.100.10.7
set routing-options autonomous-system 65000
set protocols bgp group eBGPv6 type external
set protocols bgp group eBGPv6 export BGP_export
set protocols bgp group eBGPv6 peer-as 65500
set protocols bgp group eBGPv6 neighbor 2001:db8:67::2
set protocols bgp group eBGPv4 type external
set protocols bgp group eBGPv4 export BGP_export
set protocols bgp group eBGPv4 peer-as 65500
set protocols bgp group eBGPv4 neighbor 192.168.67.2
set protocols isis interface all level 2 disable
```

Configuring Device R1

Step-by-Step Procedure

To configure a static SR-TE policy for an SRV6 tunnel over an IS-IS SRv6 core, perform the following steps on the R1 device:

1. Configure the device interfaces to enable IP transport.

```
[edit]
user@R1#set interfaces ge-0/0/0 unit 0 description R1_To_CE0
user@R1#set interfaces ge-0/0/0 unit 0 family inet address 192.168.10.2/24
user@R1#set interfaces ge-0/0/0 unit 0 family iso
user@R1#set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10::2/64
user@R1#set interfaces ge-0/0/3 unit 0 description R1_To_R2
user@R1#set interfaces ge-0/0/3 unit 0 family iso
user@R1#set interfaces ge-0/0/3 unit 0 family inet6 address 2001:db8:12::1/64
user@R1#set interfaces ge-0/0/5 unit 0 description R1_To_R4
user@R1#set interfaces ge-0/0/5 unit 0 family iso
user@R1#set interfaces ge-0/0/5 unit 0 family inet6 address 2001:db8:14::1/64
```

2. Configure the loopback interface with IPv4 and IPv6 addresses that is used as router ID for BGP sessions.

```
[edit]
user@R1#set interfaces lo0 unit 0 family iso address 49.0001.0001.0101.0100
user@R1#set interfaces lo0 unit 0 family inet6 address 2001:db8:1:255::1/128
```

3. Configure the router ID and autonomous system (AS) number to propagate routing information within a set of routing devices that belong to the same AS.

```
[edit]
user@R1#set routing-options router-id 192.168.255.11
user@R1#set routing-options autonomous-system 65500
```

4. Configure BGP on the core-facing interface to establish internal and external peering sessions.

```
[edit]
user@R1#set protocols bgp group to_R6_ibgpv6 type internal
user@R1#set protocols bgp group to_R6_ibgpv6 local-address 2001:db8:1:255::1
user@R1#set protocols bgp group to_R6_ibgpv6 import v4vpn1_res_map1
user@R1#set protocols bgp group to_R6_ibgpv6 import v6vpn1_res_map1
user@R1#set protocols bgp group to_R6_ibgpv6 family inet unicast extended-nexthop
user@R1#set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast extended-nexthop
user@R1#set protocols bgp group to_R6_ibgpv6 neighbor 2001:db8:6:255::6
```

5. Configure an external routing instance to_CE0 for both IPv4 and IPv6 traffic. Configure the BGP protocol for to_CE0 to enable peering and traffic transport between the provider edge devices.

```
[edit]
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v6 type external
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v6 as-override
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v6 peer-as 65000
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v6 neighbor 2001:db8:10::1
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v4 type external
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v4 as-override
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v4 peer-as 65000
user@R1#set routing-instances to_CE0 protocols bgp group to_CE0_v4 neighbor 192.168.10.1
```

6. Configure the resolution-map map1 with ip-color mode. Configure the BGP protocol to use multiple paths and define a policy mpath-resolve that includes the multipath-resolve action and import the policy to resolve all the available paths of IBGP multipath route.

```
[edit]
user@R1#set protocols bgp multipath
user@R1#set policy-options resolution-map map1 mode ip-color
user@R1#set policy-options policy-statement mpath-resolve then multipath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn-inet6.0 resolution-ribs inet6.3
user@R1#set routing-options resolution rib bgp.l3vpn-inet6.0 inet6-resolution-ribs inet6.3
user@R1#set routing-options resolution rib bgp.l3vpn-inet6.0 import mpath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn-inet6.0 inet6-import mpath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn-inet6.0 inet6color-import mpath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn.0 import mpath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn.0 inet6-import mpath-resolve
user@R1#set routing-options resolution rib bgp.l3vpn.0 inet6color-import mpath-resolve
user@R1#set routing-options resolution rib inet6.0 import mpath-resolve
user@R1#set routing-options resolution rib inet.0 import mpath-resolve
```

7. Configure an import and export policy for the R1 device's VRF table.

```
[edit]
user@R1#set policy-options policy-statement to_CE0_community_import term 0 from community
to_CE0_community
user@R1#set policy-options policy-statement to_CE0_community_import term 0 then accept
user@R1#set policy-options policy-statement to_CE0_community_export term 0 then community
add to_CE0_community
```

```
user@R1#set policy-options policy-statement to_CE0_community_export term 0 then next-hop
2001:db8:0:a1::d01
user@R1#set policy-options policy-statement to_CE0_community_export term 0 then accept
user@R1#set routing-instances to_CE0 vrf-import to_CE0_community_import
user@R1#set routing-instances to_CE0 vrf-export to_CE0_community_export
```

8. Configure the VPN type and a unique route distinguisher for each PE router participating in the routing instance.

```
[edit]
user@R1#set routing-instances to_CE0 instance-type vrf
user@R1#set routing-instances to_CE0 interface ge-0/0/0.0
user@R1#set routing-instances to_CE0 route-distinguisher 192.168.255.11:1
```

9. Define a policy to load-balance packets and apply the per-packet policy to enable load balancing of traffic.

```
[edit]
user@R1#set policy-options policy-statement LBPP term 1 then load-balance per-packet
user@R1#set policy-options community to_CE0_community members target:65500:1
user@R1#set routing-options forwarding-table export LBPP
```

10. Define a policy v4vpn1_res_map1 and v6vpn1_res_map1 to accept the routes advertised from R1.

```
[edit]
user@R1#set policy-options policy-statement v4vpn1_res_map1 term 1 from protocol bgp
user@R1#set policy-options policy-statement v4vpn1_res_map1 term 1 then accept
user@R1#set policy-options policy-statement v4vpn1_res_map1 term 1 then resolution-map map1
user@R1#set policy-options policy-statement v6vpn1_res_map1 term 1 from family inet6-vpn
user@R1#set policy-options policy-statement v6vpn1_res_map1 term 1 then accept
user@R1#set policy-options policy-statement v6vpn1_res_map1 term 1 then resolution-map map1
```

11. Disable level 2, enable IS-IS as the interior gateway protocol (IGP) for routing traffic between the core devices.

```
[edit]
user@R1#set protocols isis interface all level 2 disable
```

```
user@R1#set protocols isis interface fxp0.0 disable
user@R1#set protocols isis interface lo0.0
```

12. Enable TI-LFA for the IS-IS protocol.

```
[edit]
user@R1#set protocols isis backup-spf-options use-post-convergence-lfa
user@R1#set protocols isis backup-spf-options use-source-packet-routing
```

13. Configure the IPv6 index value of the node segment.

```
[edit]
user@R1#set protocols isis source-packet-routing node-segment ipv6-index 101
```

14. Enable SRv6 globally and the locator address to indicate the SRv6 capability of the router. SRv6 SID is an IPv6 address that consists of the locator and a function. The routing protocols advertise the locator addresses.

```
[edit]
user@R1#set protocols source-packet-routing srv6
user@R1#set routing-options source-packet-routing srv6 locator loc1 2001:db8:0:a1::/112
```

NOTE: Starting in Junos OS Release 22.3R1, the locator length is limited to a maximum value of 112, so that the remaining 16 bits would be used as function length. The locator configuration is extended with the configuration of block length, function length and static function max entries. Hence, some static SID format may fail the commit check. The static SIDS within the SRv6 locator can only range from fec0:6bfd:1ba:20:0001:0:0:0 to fec0:6bfd:1ba:20:7fff:0:0:0.

15. Enable preserve nexthop hierarchy for SR-TE route flavors and enable platform merge for SRv6 chain nexthops.

```
[edit]
user@R1#set routing-options resolution preserve-nexthop-hierarchy
user@R1#set routing-options forwarding-table srv6-chain-merge
```

16. Configure the end-dt4 and end-dt6 SID values for enabling the Layer 3 VPN services.

```
[edit]
user@R1#set routing-instances to_CE0 protocols bgp source-packet-routing srv6 locator loc1
end-dt4-sid 2001:db8:0:a1::d410
user@R1#set routing-instances to_CE0 protocols bgp source-packet-routing srv6 locator loc1
end-dt6-sid 2001:db8:0:a1::d610
```

17. Enable the device to advertise the SRv6 services to BGP peers and to accept the routes advertised by the egress devices.

```
[edit]
user@R1#set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast advertise-srv6-service
user@R1#set protocols bgp group to_R6_ibgpv6 family inet-vpn unicast accept-srv6-service
user@R1#set protocols bgp group to_R6_ibgpv6 family inet6 unicast extended-nexthop-color
user@R1#set protocols bgp group to_R6_ibgpv6 family inet6-vpn unicast advertise-srv6-service
user@R1#set protocols bgp group to_R6_ibgpv6 family inet6-vpn unicast accept-srv6-service
```

18. Configure the End-Sid function for the prefix segments. Specify a flavor, that is the behavior of the End-SID function as per your network requirements. Penultimate Segment Pop (PSP), Ultimate Segment Pop (USP), and Ultimate Segment Decapsulation (USP) are the three available flavors for SRv6 functions.

NOTE: Ensure that the locator and the End-SID are in the same subnet to avoid a commit error.

```
[edit]
user@R1#set protocols isis source-packet-routing srv6 locator loc1 end-sid
2001:db8:0:a1::d01 flavor psp
user@R1#set protocols isis source-packet-routing srv6 locator loc1 end-sid
2001:db8:0:a1::d01 flavor usp
user@R1#set protocols isis source-packet-routing srv6 locator loc1 end-sid
2001:db8:0:a1::d01 flavor usd
```

19. Configure End-X-SID function on the point-to-point (P2P) interface for the adjacency segments. Specify one or more flavor for the End-X-SID.

NOTE: Ensure that the Locator and End-X-SID are in the same subnet to avoid a commit error. You must enable SRv6 and configure the locator at the `[edit routing-options]` before mapping locators to interfaces.

```
[edit]
user@R1#set protocols isis interface ge-0/0/3.0 level 2 disable
user@R1#set protocols isis interface ge-0/0/3.0 level 1 srv6-adjacency-segment unprotected
locator loc1 end-x-sid 2001:db8:0:a1::1a12 flavor psp
user@R1#set protocols isis interface ge-0/0/3.0 level 1 srv6-adjacency-segment unprotected
locator loc1 end-x-sid 2001:db8:0:a1::1a12 flavor usd
user@R1#set protocols isis interface ge-0/0/3.0 point-to-point
user@R1#set protocols isis interface ge-0/0/5.0 level 2 disable
user@R1#set protocols isis interface ge-0/0/5.0 level 1 post-convergence-lfa
user@R1#set protocols isis interface ge-0/0/5.0 point-to-point
```

20. Configure the SRv6 segment lists end-sids-segment and end-x-sids-segment-last-sid-end-sid between <R4, R5, R6> and <R2, R3, R6>.

```
[edit]
user@R1#set protocols source-packet-routing segment-list end-sids-segment srv6
user@R1#set protocols source-packet-routing segment-list end-sids-segment hop1 srv6-sid
2001:db8:0:a4::d04
user@R1#set protocols source-packet-routing segment-list end-sids-segment hop2 srv6-sid
2001:db8:0:a5::d05
user@R1#set protocols source-packet-routing segment-list end-sids-segment hop3 srv6-sid
2001:db8:0:a6::d06
user@R1#set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-
sid srv6
user@R1#set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-
sid hop1 srv6-sid 2001:db8:0:a2::1a23
user@R1#set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-
sid hop2 srv6-sid 2001:db8:0:a3::1a34
user@R1#set protocols source-packet-routing segment-list end-x-sids-segment-last-sid-end-
sid hop3 srv6-sid 2001:db8:0:a6::d06
```

21. Configure the SRv6-TE tunnel between R1 and R6 with end-sids-segment weight 40 and end-x-sids-segment-last-sid-end-sid weight 30 for uncolored paths (nc_path_R1R6) and colored paths (c_path_R1R6).

```
[edit]
user@R1#set protocols source-packet-routing source-routing-path nc_path_R1R6 srv6
user@R1#set protocols source-packet-routing source-routing-path nc_path_R1R6 to
2001:db8:0:a6::d06
user@R1#set protocols source-packet-routing source-routing-path nc_path_R1R6 from
2001:db8:1:255::1
user@R1#set protocols source-packet-routing source-routing-path nc_path_R1R6 primary end-
sids-segment weight 40
user@R1#set protocols source-packet-routing source-routing-path nc_path_R1R6 primary end-x-
sids-segment-last-sid-end-sid weight 30
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 srv6
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 to
2001:db8:0:a6::d06
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 from
2001:db8:1:255::1
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 color 6
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 primary end-
sids-segment weight 40
user@R1#set protocols source-packet-routing source-routing-path c_path_R1R6 primary end-x-
sids-segment-last-sid-end-sid weight 30
```

Results

Check the results of the configuration:

```
interfaces {
    ge-0/0/0 {
        unit 0 {
            description R1_To_CE0;
            family inet {
                address 192.168.10.1/24;
                address 192.168.10.2/24;
            }
            family iso;
            family inet6 {
                address 2001:db8:10::2/64;
            }
```

```
        }
    }
    ge-0/0/3 {
        unit 0 {
            description R1_To_R2;
            family iso;
            family inet6 {
                address 2001:db8:12::1/64;
            }
        }
    }
    ge-0/0/5 {
        unit 0 {
            description R1_To_R4;
            family iso;
            family inet6 {
                address 2001:db8:14::1/64;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.100.2/32;
            }
            family iso {
                address 49.0002.0192.0168.0002.00;
                address 49.0001.0001.0101.0100;
            }
            family inet6 {
                address 2001:db8:1:255::1/128;
            }
        }
    }
}
policy-options {
    policy-statement LBPP {
        term 1 {
            then {
                load-balance per-packet;
            }
        }
    }
```

```
policy-statement mpath-resolve {
    then multipath-resolve;
}
policy-statement to_CE0_community_export {
    term 0 {
        then {
            community add to_CE0_community;
            next-hop 2001:db8:0:a1::d01;
            accept;
        }
    }
}
policy-statement to_CE0_community_import {
    term 0 {
        from community to_CE0_community;
        then accept;
    }
}
policy-statement v4vpn1_res_map1 {
    term 1 {
        from protocol bgp;
        then {
            accept;
            resolution-map map1;
        }
    }
}
policy-statement v6vpn1_res_map1 {
    term 1 {
        from {
            family inet6-vpn;
            protocol bgp;
        }
        then {
            accept;
            resolution-map map1;
        }
    }
}
community to_CE0_community members target:65500:1;
resolution-map map1 {
    mode ip-color;
}
```

```
}
routing-instances {
    to_CE0 {
        instance-type vrf;
        protocols {
            bgp {
                group to_CE0_v6 {
                    type external;
                    as-override;
                    peer-as 65000;
                    neighbor 2001:db8:10::1;
                }
                group to_CE0_v4 {
                    type external;
                    as-override;
                    peer-as 65000;
                    neighbor 192.168.10.1;
                }
                source-packet-routing {
                    srv6 {
                        locator loc1 {
                            end-dt4-sid 2001:db8:0:a1::d410;
                            end-dt6-sid 2001:db8:0:a1::d610;
                        }
                    }
                }
            }
        }
        interface ge-0/0/0.0;
        route-distinguisher 192.168.255.11:1;
        vrf-import to_CE0_community_import;
        vrf-export to_CE0_community_export;
    }
}
routing-options {
    source-packet-routing {
        srv6 {
            locator loc1 2001:db8:0:a1::/64;
        }
    }
    resolution {
        preserve-nexthop-hierarchy;
        rib bgp.l3vpn-inet6.0 {
```

```
            resolution-ribs inet6.3;
            inet6-resolution-ribs inet6.3;
            import mpath-resolve;
            inet6-import mpath-resolve;
            inet6color-import mpath-resolve;
        }
        rib bgp.l3vpn.0 {
            import mpath-resolve;
            inet6-import mpath-resolve;
            inet6color-import mpath-resolve;
        }
        rib inet6.0 {
            import mpath-resolve;
        }
        rib inet.0 {
            import mpath-resolve;
        }
    }
    router-id 192.168.255.11;
    autonomous-system 65500;
    forwarding-table {
        srv6-chain-merge;
        export LBPP;
    }
}
protocols {
    bgp {
        group to_R6_ibgpv6 {
            type internal;
            local-address 2001:db8:1:255::1;
            import [ v4vpn1_res_map1 v6vpn1_res_map1 ];
            family inet {
                unicast {
                    extended-nexthop;
                }
            }
            family inet-vpn {
                unicast {
                    extended-nexthop;
                    advertise-srv6-service;
                    accept-srv6-service;
                }
            }
```

```
            family inet6 {
                unicast {
                    extended-nexthop-color;
                }
            }
            family inet6-vpn {
                unicast {
                    advertise-srv6-service;
                    accept-srv6-service;
                }
            }
            neighbor 2001:db8:6:255::6;
        }
        multipath;
    }
    isis {
        interface ge-0/0/3.0 {
            level 2 disable;
            level 1 {
                srv6-adjacency-segment {
                    unprotected {
                        locator loc1 {
                            end-x-sid 2001:db8:0:a1::1a12 {
                                flavor {
                                    psp;
                                    usd;
                                }
                            }
                        }
                    }
                }
            }
            point-to-point;
        }
        interface ge-0/0/5.0 {
            level 2 disable;
            level 1 {
                post-convergence-lfa;
            }
            point-to-point;
        }
        interface all {
            level 2 disable;
```

```
        }
        interface fxp0.0 {
            disable;
        }
        interface lo0.0;
        source-packet-routing {

            node-segment ipv6-index 101;
            srv6 {
                locator loc1 {
                    end-sid 2001:db8:0:a1::d01 {
                        flavor {
                            psp;
                            usp;
                            usd;
                        }
                    }
                }
            }
        }
        backup-spf-options {
            use-post-convergence-lfa;
            use-source-packet-routing;
        }
    }
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    source-packet-routing {
        segment-list end-sids-segment {
            srv6;
            hop1 srv6-sid 2001:db8:0:a4::d04;
            hop2 srv6-sid 2001:db8:0:a5::d05;
            hop3 srv6-sid 2001:db8:0:a6::d06;
        }
        segment-list end-x-sids-segment-last-sid-end-sid {
            srv6;
            hop1 srv6-sid 2001:db8:0:a2::1a23;
            hop2 srv6-sid 2001:db8:0:a3::1a34;
            hop3 srv6-sid 2001:db8:0:a6::d06;
```

```
        }
        srv6;
        source-routing-path nc_path_R1R6 {
            srv6;
            to 2001:db8:0:a6::d06;
            from 2001:db8:1:255::1;
            primary {
                end-sids-segment weight 40;
                end-x-sids-segment-last-sid-end-sid weight 30;
            }
        }
        source-routing-path c_path_R1R6 {
            srv6;
            to 2001:db8:0:a6::d06;
            from 2001:db8:1:255::1;
            color 6;
            primary {
                end-sids-segment weight 40;
                end-x-sids-segment-last-sid-end-sid weight 30;
            }
        }
    }
}
```

When done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

- Verifying SPRING traffic-engineered LSP | **980**
- Verifying Transport RIB populated by SR-TE | **980**
- Verifying BGP Service IPv4 route over uncolored SR-TE SRv6 route End.DT4 | **986**
- Verifying BGP Service IPv6 route over colored SR-TE SRv6 route End.DT6 | **987**
- Verifying IPv4 Connectivity Between CE0 and CE7 | **988**

Confirm that the configuration is working properly.

Verifying SPRING traffic-engineered LSP

Purpose

Verifying SPRING traffic-engineered LSP on the ingress device R1

Action

From operational mode, run the **show spring-traffic-engineering lsp** command on the device R1.

```
user@R1>show spring-traffic-engineering lsp
To              State     LSPname
2001:db8:0:a6::d06-6<c6> Up c_path_R1R6
2001:db8:0:a6::d06 Up     nc_path_R1R6
```

Meaning

The output displays the SPRING traffic-engineered LSPs on the ingress device.

Verifying Transport RIB populated by SR-TE

Purpose

Verifying Transport RIB populated by SR-TE.

Action

From operational mode, run the **show route protocol spring-te extensive** command on the device R1.

```
user@R1>show route protocol spring-te extensive
inet.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)

to_CE0.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)

iso.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
```

```
inet6.0: 36 destinations, 36 routes (36 active, 0 holddown, 0 hidden)

inet6.3: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
2001:db8:0:a6::d06/128 (2 entries, 1 announced)
        *SPRING-TE Preference: 8
                Next hop type: Indirect, Next hop index: 0
                Address: 0x7972548
                Next-hop reference count: 3
                Next hop type: Chain, Next hop index: 0
        Next hop: via Chain Tunnel Composite, SRv6
        Next hop: ELNH Address 0x76b7aa8, selected
        SRV6-Tunnel: Reduced-SRH Encap-mode
         Src: 2001:db8:1:255::1 Dest: 2001:db8:0:a6::d06
         Segment-list[0] 2001:db8:0:a2::1a23
         Segment-list[1] 2001:db8:0:a3::1a34
         Segment-list[2] 2001:db8:0:a6::d06
            Next hop type: Chain, Next hop index: 0
            Address: 0x76b7aa8
            Next-hop reference count: 5
            Next hop: via Chain Tunnel Composite, SRv6
            Next hop: ELNH Address 0x76b7a3c
            SRV6-Tunnel: Reduced-SRH Encap-mode
             Src: abcd::128:205:174:232 Dest: 2001:db8:0:a2::
             Segment-list[0] 2001:db8:0:a2::
                Next hop type: Router, Next hop index: 634
                Address: 0x76b7a3c
                Next-hop reference count: 17
                Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0 weight 0x1
                Next hop type: Chain, Next hop index: 0
        Next hop: via Chain Tunnel Composite, SRv6
        Next hop: ELNH Address 0x797282c, selected
        SRV6-Tunnel: Reduced-SRH Encap-mode
         Src: 2001:db8:1:255::1 Dest: 2001:db8:0:a6::d06
         Segment-list[0] 2001:db8:0:a4::d04
         Segment-list[1] 2001:db8:0:a5::d05
         Segment-list[2] 2001:db8:0:a6::d06
            Next hop: ELNH Address 0x76b9104 weight 0x1, selected
                Next hop type: Chain, Next hop index: 0
                Address: 0x76b9104
                Next-hop reference count: 1
                Next hop: via Chain Tunnel Composite, SRv6
                Next hop: ELNH Address 0x76b8ee8
```

```
                SRV6-Tunnel: Reduced-SRH Encap-mode
                 Src: abcd::128:205:174:232 Dest: 2001:db8:0:a4::
                 Segment-list[0] 2001:db8:0:a4::
                    Next hop type: Router, Next hop index: 635
                    Address: 0x76b8ee8
                    Next-hop reference count: 32
                    Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0 weight 0x1
            Next hop: ELNH Address 0x76b9170 weight 0xf000
                Next hop type: Chain, Next hop index: 0
                Address: 0x76b9170
                Next-hop reference count: 1
                Next hop: via Chain Tunnel Composite, SRv6
                Next hop: ELNH Address 0x76b8f54
                SRV6-Tunnel: Reduced-SRH Encap-mode
                 Src: abcd::128:205:174:232 Dest: 2001:db8:0:a4::
                 Segment-list[0] 2001:db8:0:a3::d03
                 Segment-list[1] 2001:db8:0:a4::
                    Next hop type: Router, Next hop index: 634
                    Address: 0x76b8f54
                    Next-hop reference count: 11
                    Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0 weight 0xf000
                Protocol next hop: 2001:db8:0:a2::1a23 Balance: 43%
                Indirect next hop: 0x7165534 - INH Session ID: 0 Weight 0x1
                Protocol next hop: 2001:db8:0:a4::d04 Balance: 57%
                Indirect next hop: 0x71656cc - INH Session ID: 0 Weight 0x1
                State: <Active Int>
                Local AS: 65500
                Age: 14:29:23   Metric: 1       Metric2: 30
                Validation State: unverified
                Task: SPRING-TE
                Announcement bits (5): 0-Resolve tree 2 2-Resolve tree 4 3-Resolve tree 6 4-
Resolve_IGP_FRR task 5-Resolve tree 10
                AS path: I
                SRTE Policy State:
                   SR Preference/Override: 100/100
                   Tunnel Source: Static configuration
                Session-IDs associated:
                Session-id: 325 Version: 1
                Session-id: 327 Version: 1
                Thread: junos-main
                Indirect next hops: 2
                        Protocol next hop: 2001:db8:0:a2::1a23 Metric: 10
                        Indirect next hop: 0x7165534 - INH Session ID: 0 Weight 0x1
```

```
                        Indirect path forwarding next hops: 1
                                Next hop type: Chain
                                Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0
                                2001:db8:0:a2::/64 Originating RIB: inet6.3
                                  Metric: 10 Node path count: 1
                                  Forwarding nexthops: 1
                                        Next hop type: Chain
                                        Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0
                        Protocol next hop: 2001:db8:0:a4::d04 Metric: 10
                        Indirect next hop: 0x71656cc - INH Session ID: 0 Weight 0x1
                        Indirect path forwarding next hops: 1
                                Next hop type: Chain
                                Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0
                                fe80::5668:acff:feda:cc1b via ge-0/0/3.0
                                2001:db8:0:a4::/64 Originating RIB: inet6.3
                                  Metric: 10 Node path count: 1
                                  Forwarding nexthops: 2
                                        Next hop type: List
                                        Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0
                                        Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0

to_CE0.inet6.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)

bgp.l3vpn-inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6color.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

2001:db8:0:a6::d06-6<c6>/160 (1 entry, 1 announced)
        *SPRING-TE Preference: 8
                Next hop type: Indirect, Next hop index: 0
                Address: 0x79724b4
                Next-hop reference count: 1
                Next hop type: Chain, Next hop index: 0
        Next hop: via Chain Tunnel Composite, SRv6
        Next hop: ELNH Address 0x76b7aa8, selected
        SRV6-Tunnel: Reduced-SRH Encap-mode
         Src: 2001:db8:1:255::1 Dest: 2001:db8:0:a6::d06-6<c6>
         Segment-list[0] 2001:db8:0:a2::1a23
         Segment-list[1] 2001:db8:0:a3::1a34
         Segment-list[2] 2001:db8:0:a6::d06
            Next hop type: Chain, Next hop index: 0
            Address: 0x76b7aa8
            Next-hop reference count: 5
```

```
            Next hop: via Chain Tunnel Composite, SRv6
            Next hop: ELNH Address 0x76b7a3c
            SRV6-Tunnel: Reduced-SRH Encap-mode
             Src: abcd::128:205:174:232 Dest: 2001:db8:0:a2::
             Segment-list[0] 2001:db8:0:a2::
                Next hop type: Router, Next hop index: 634
                Address: 0x76b7a3c
                Next-hop reference count: 17
                Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0 weight 0x1
                Next hop type: Chain, Next hop index: 0
        Next hop: via Chain Tunnel Composite, SRv6
        Next hop: ELNH Address 0x797282c, selected
        SRV6-Tunnel: Reduced-SRH Encap-mode
         Src: 2001:db8:1:255::1 Dest: 2001:db8:0:a6::d06-6<c6>
         Segment-list[0] 2001:db8:0:a4::d04
         Segment-list[1] 2001:db8:0:a5::d05
         Segment-list[2] 2001:db8:0:a6::d06
           Next hop: ELNH Address 0x76b9104 weight 0x1, selected
                Next hop type: Chain, Next hop index: 0
                Address: 0x76b9104
                Next-hop reference count: 1
                Next hop: via Chain Tunnel Composite, SRv6
                Next hop: ELNH Address 0x76b8ee8
                SRV6-Tunnel: Reduced-SRH Encap-mode
                 Src: abcd::128:205:174:232 Dest: 2001:db8:0:a4::
                 Segment-list[0] 2001:db8:0:a4::
                    Next hop type: Router, Next hop index: 635
                    Address: 0x76b8ee8
                    Next-hop reference count: 32
                    Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0 weight 0x1
           Next hop: ELNH Address 0x76b9170 weight 0xf000
                Next hop type: Chain, Next hop index: 0
                Address: 0x76b9170
                Next-hop reference count: 1
                Next hop: via Chain Tunnel Composite, SRv6
                Next hop: ELNH Address 0x76b8f54
                SRV6-Tunnel: Reduced-SRH Encap-mode
                 Src: abcd::128:205:174:232 Dest: 2001:db8:0:a4::
                 Segment-list[0] 2001:db8:0:a3::d03
                 Segment-list[1] 2001:db8:0:a4::
              Next hop type: Router, Next hop index: 634
                    Address: 0x76b8f54
                    Next-hop reference count: 11
```

```
    Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0 weight 0xf000
Protocol next hop: 2001:db8:0:a2::1a23 Balance: 43%
Indirect next hop: 0x716539c - INH Session ID: 0 Weight 0x1
Protocol next hop: 2001:db8:0:a4::d04 Balance: 57%
Indirect next hop: 0x7165864 - INH Session ID: 0 Weight 0x1
State: <Active Int>
Local AS: 65500
Age: 14:29:23   Metric: 1       Metric2: 30
Validation State: unverified
Task: SPRING-TE
Announcement bits (1): 1-Resolve tree 11
AS path: I
SRTE Policy State:
   SR Preference/Override: 100/100
   Tunnel Source: Static configuration
Thread: junos-main
Indirect next hops: 2
        Protocol next hop: 2001:db8:0:a2::1a23 Metric: 10
        Indirect next hop: 0x716539c - INH Session ID: 0 Weight 0x1
        Indirect path forwarding next hops: 1
                Next hop type: Chain
                Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0
                2001:db8:0:a2::/64 Originating RIB: inet6.3
                  Metric: 10 Node path count: 1
                  Forwarding nexthops: 1
                        Next hop type: Chain
                        Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0
        Protocol next hop: 2001:db8:0:a4::d04 Metric: 10
        Indirect next hop: 0x7165864 - INH Session ID: 0 Weight 0x1
        Indirect path forwarding next hops: 1
                Next hop type: Chain
                Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0
                fe80::5668:acff:feda:cc1b via ge-0/0/3.0
                2001:db8:0:a4::/64 Originating RIB: inet6.3
                  Metric: 10 Node path count: 1
                  Forwarding nexthops: 2
                        Next hop type: List
                        Next hop: fe80::5668:acff:feda:cc51 via ge-0/0/5.0
                        Next hop: fe80::5668:acff:feda:cc1b via ge-0/0/3.0
```

Meaning

The output displays colored and uncolored SR-TE transport routes, with each route having three SRv6-TE segment-lists. The output also signifies that the colored and uncolored routes segment-lists follow reduced SRH encapsulation mode.

Verifying BGP Service IPv4 route over uncolored SR-TE SRv6 route End.DT4

Purpose

Verify the BGP Service IPv4 route resolves over uncolored SR-TE SRv6 route End.DT4

Action

From operational mode, run the **show route 10.100.10.7 extensive expanded-nh** command on the device R1.

```
user@R1>show route 10.100.10.7 extensive expanded-nh
to_CE0.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)10.100.10.7/32 (1 entry,
1 announced)
Installed-nexthop:
Indr Composite (0x76ba328) 2001:db8:0:a6::d06 Session-ID: 327
  Krt_cnh (0x6fb4328) Index:642
    Krt_inh (0x7166854) Index:1048583 PNH: 2001:db8:0:a6::d06  SRv6-TE uncolored LSP
      List (0x7972f1c) Index:1048582
        Frr_inh (0x76ba10c) Index:1048577 PNH: 2001:db8:0:a2::1a23 Session-ID: 324
          Chain Fully resolved tunnel (0x76b7cc4) Index:637 SRv6
            Router (0x76b7a3c) Index:634 fe80::5668:acff:feda:cc1b Session-ID: 322 via ge-0/0/3.0
        Frr_inh (0x76b9fc8) Index:1048580 PNH: 2001:db8:0:a4::d04 Session-ID: 326
          List (0x7972a7c) Index:1048578
            Chain Fully resolved tunnel (0x76b8d38) Index:638 SRv6
              Router (0x76b8ee8) Index:635 fe80::5668:acff:feda:cc51 Session-ID: 323 via
ge-0/0/5.0
            Chain Fully resolved tunnel (0x76b9464) Index:639 SRv6
              Router (0x76b8f54) Index:634 fe80::5668:acff:feda:cc1b Session-ID: 322 via
ge-0/0/3.0
TSI:
KRT in-kernel 10.100.10.7/32 -> {composite(642)}
        *BGP    Preference: 170/-101
                Route Distinguisher: 192.168.255.66:6
                Next hop type: Indirect, Next hop index: 0
                Address: 0x76ba250
```

```
                Next-hop reference count: 6
                Source: 2001:db8:6:255::6
                Next hop type: Chain, Next hop index: 0
```

Meaning

The output confirms that the BGP VPN IPv4 service prefix 10.100.10.7/32 is installed in the vpn.inet.0 table that resolves over uncolored SRv6-TE policy.

Verifying BGP Service IPv6 route over colored SR-TE SRv6 route End.DT6

Purpose

Verify that the BGP VPN IPv6 service route resolves over colored SRv6-TE policy.

Action

From operational mode, run the **show route 2001:db8:7:255::7/128 extensive expanded-nh** command on the device R1.

```
user@R1>show route 2001:db8:7:255::7/128 extensive expanded-nh
to_CE0.inet6.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
2001:db8:7:255::7/128 (1 entry, 1 announced)
Installed-nexthop:
Indr Composite (0x76ba1e4) 2001:db8:0:a6::d06-6<c6>
  Krt_cnh (0x6fb25f4) Index:647
    Krt_inh (0x7166d1c) Index:1048586 PNH: 2001:db8:0:a6::d06-6<c6>  SRv6-TE IPV6 colored LSP
      List (0x7972f1c) Index:1048585
        Frr_inh (0x76ba034) Index:1048577 PNH: 2001:db8:0:a2::1a23 Session-ID: 328
          Chain Fully resolved tunnel (0x76b7bec) Index:640 SRv6
            Router (0x76b7a3c) Index:634 fe80::5668:acff:feda:cc1b Session-ID: 322 via ge-0/0/3.0
        Frr_inh (0x76b9f5c) Index:1048582 PNH: 2001:db8:0:a4::d04 Session-ID: 329
          List (0x79729e8) Index:1048581
            Chain Fully resolved tunnel (0x76b938c) Index:641 SRv6
              Router (0x76b8ee8) Index:635 fe80::5668:acff:feda:cc51 Session-ID: 323 via
ge-0/0/5.0
            Chain Fully resolved tunnel (0x76b93f8) Index:642 SRv6
              Router (0x76b8f54) Index:634 fe80::5668:acff:feda:cc1b Session-ID: 322 via
ge-0/0/3.0
TSI:
```

```
KRT in-kernel 2001:db8:7:255::7/128 -> {composite(647)}
        *BGP    Preference: 170/-101
                Route Distinguisher: 192.168.255.66:6
                Next hop type: Indirect, Next hop index: 0
                Address: 0x76ba394
                Next-hop reference count: 3
                Source: 2001:db8:6:255::6
                Next hop type: Chain, Next hop index: 0
        Next hop: via Chain Tunnel Composite, SRv6
        Next hop: ELNH Address 0x76b7aa8, selected
        SRV6-Tunnel: Reduced-SRH Encap-mode
         Src: 2001:db8:1:255::1 Dest: 2001:db8:0:a6::d06-6<c6>
         Segment-list[0] 2001:db8:0:a2::1a23
         Segment-list[1] 2001:db8:0:a3::1a34
         Segment-list[2] 2001:db8:0:a6::d06
            Next hop type: Chain, Next hop index: 0
            Address: 0x76b7aa8
            Next-hop reference count: 5
```

Meaning

The output confirms that the BGP VPN IPv6 service prefix 2001:db8:7:255::7/128 is installed in the vpn.inet6.0 table that resolves over colored SRv6-TE policy.

Verifying IPv4 Connectivity Between CE0 and CE7

Purpose

Generate pings to verify IPv4 connectivity between the CE devices over the IPv6 provider core.

Action

From operational mode, run the **ping 10.100.10.7** command on the device CE0.

```
user@CE0> ping 10.100.10.7
PING 10.100.10.7 (10.100.10.7): 56 data bytes
64 bytes from 10.100.10.7: icmp_seq=0 ttl=62 time=9.363 ms
64 bytes from 10.100.10.7: icmp_seq=1 ttl=62 time=7.696 ms
^C
--- 10.100.10.7 ping statistics ---
```

```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 7.696/8.529/9.363/0.834 ms
```

Meaning

The output confirms IPv4 connectivity is working between the CE device networks. This provides verification that SRv6 tunneling over an IPv6 provider core is working properly in this example.

Change History Table

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

Release	Description
Junos OS Release 20.2R1	Starting in Junos OS Release 20.2R1, Junos OS provides support for controller based BGP-SRTE routes are installed as segment routing traffic-engineered (SPRING-TE) routes
18.3R1	Starting in Release 18.3R1, Junos OS supports collection of traffic statistics for both ingress IP and transit MPLS traffic in a network configured with segment routing traffic engineering policy. To enable collection of traffic statistics include the `telemetry` statement at the `[edit protocols source-packet-routing]` hierarchy level.
change-completed	

Link-State Distribution Using BGP

IN THIS SECTION

- Link-State Distribution Using BGP Overview | **990**
- Example: Configuring Link State Distribution Using BGP | **1005**
- Configuring Link State Distribution Using BGP | **1029**
- Link-State Distribution using SRv6 | **1033**
- IPv6 Prefixes and IPv6 Adjacency SIDs MPLS Support in Traffic Engineering Database and BGP Link-State | **1036**

Link-State Distribution Using BGP Overview

IN THIS SECTION

- Role of an Interior Gateway Protocol | **990**
- Limitations of an Interior Gateway Protocol | **991**
- Need for Spanning Link-State Distribution | **991**
- Using BGP as a Solution | **992**
- Supported and Unsupported Features | **998**
- BGP Link-State Extensions for Source Packet Routing in Networking (SPRING) | **999**
- Verifying NLRI Node Learned Through BGP with OSPF as IGP | **1002**
- Verifying the Prefix NLRI Learned Through BGP with OSPF as IGP | **1003**

Role of an Interior Gateway Protocol

An interior gateway protocol (IGP) is a type of protocol used for exchanging routing information between devices within an autonomous system (AS). Based on the method of computing the best path to a destination, the IGPs are divided into two categories:

- Link-state protocols—Advertise information about the network topology (directly connected links and the state of those links) to all routers using multicast addresses and triggered routing updates until all the routers running the link-state protocol have identical information about the internetwork. The best path to a destination is calculated based on constraints such as maximum delay, minimum available bandwidth, and resource class affinity.

  OSPF and IS-IS are examples of link-state protocols.

- Distance vector protocols—Advertise complete routing table information to directly connected neighbors using a broadcast address. The best path is calculated based on the number of hops to the destination network.

  RIP is an example of a distance vector protocol.

As the name implies, the role of an IGP is to provide routing connectivity within or internal to a given routing domain. A routing domain is a set of routers under common administrative control that share a common routing protocol. An AS can consist of multiple routing domains, where IGP functions to advertise and learn network prefixes (routes) from neighboring routers to build a route table that ultimately contains entries for all sources advertising reachability for a given prefix. IGP executes a route

selection algorithm to select the best path between the local router and each destination, and provides full connectivity among the routers making up a routing domain.

In addition to advertising internal network reachability, IGPs are often used to advertise routing information that is external to that IGP's routing domain through a process known as route redistribution. Route redistribution is the process of exchanging routing information among distinct routing protocols to tie multiple routing domains together when intra-AS connectivity is desired.

Limitations of an Interior Gateway Protocol

While each individual IGP has its own advantages and limitations, the biggest limitations of IGP in general are performance and scalability.

IGPs are designed to handle the task of acquiring and distributing network topology information for traffic engineering purposes. While this model has served well, IGPs have inherent scaling limitations when it comes to distributing large databases. IGPs can autodetect neighbors, with which they acquire intra-area network topology information. However, the link-state database or a traffic engineering database has the scope of a single area or AS, thereby limiting applications, such as end-to-end traffic engineering, the benefit of having external visibility to make better decisions.

For label-switched networks, such as MPLS and Generalized MPLS (GMPLS), most existing traffic engineering solutions work in a single routing domain. These solutions do not work when a route from the ingress node to the egress node leaves the routing area or AS of the ingress node. In such cases, the path computation problem becomes complicated because of the unavailability of the complete routing information throughout the network. This is because service providers usually choose not to leak routing information beyond the routing area or AS for scalability constraints and confidentiality concerns.

Need for Spanning Link-State Distribution

One of the limitations of IGP is its inability to span link-state distribution outside a single area or AS. However, spanning link-state information acquired by an IGP across multiple areas or ASs has the following needs:

- LSP path computation—This information is used to compute the path for MPLS LSPs across multiple routing domains, for example an inter-area TE LSP.
- External path computing entities—External path computing entities, such as Application Layer Traffic Optimization (ALTO) and Path Computation Elements (PCE), perform path computations based on the network topology and current state of connections within the network, including traffic engineering information. This information is typically distributed by IGPs within the network.

  However, because the external path computing entities cannot extract this information from the IGPs, they perform network monitoring to optimize network services.

Using BGP as a Solution

Overview

To meet the needs for spanning link-state distribution across multiple domains, an exterior gateway protocol (EGP) is required to collect link-state and traffic engineering information from an IGP area, share it with external component, and use it for computing paths for interdomain MPLS LSPs.

BGP is a standardized EGP designed to exchange routing and reachability information between autonomous systems (ASs). BGP is a proven protocol that has better scaling properties because it can distribute millions of entries (for example, VPN prefixes) in a scalable fashion. BGP is the only routing protocol in use today that is suited to carry all of the routes in the Internet. This is largely because BGP runs on top of TCP and can make use of TCP flow control. In contrast, the internal gateway protocols (IGPs) do not have flow control. When IGPs have too much route information, they begin to churn. When BGP has a neighboring speaker that is sending information too quickly, BGP can throttle down the neighbor by delaying TCP acknowledgments.

Another benefit of BGP is that it uses type, length, value (TLV) tuples and network layer reachability information (NLRI) that provide seemingly endless extensibility without the need for the underlying protocol to be altered.

The distribution of link-state information across domains is regulated using policies to protect the interests of the service provider. This requires a control over the topology distribution using policies. BGP with its implemented policy framework serves well in the interdomain route distribution. In Junos OS, BGP is completely policy driven. The operator must explicitly configure neighbors to peer with and explicitly accept routes into BGP. Furthermore, routing policy is used to filter and modify routing information. Thus, routing policies provide complete administrative control over the routing tables.

Although, within an AS, both IGP-TE and BGP-TE provide the same set of information, BGP-TE has better scaling characteristics that are inherited from the standard BGP protocol. This makes BGP-TE a more scalable choice for acquiring multi-area/multi-AS topology information.

By using BGP as a solution, the IGP-acquired information is used for distribution into BGP. The ISPs can selectively expose this information with other ISPs, service providers, and content distribution networks (CDNs) through normal BGP peering. This allows for aggregation of the IGP-acquired information across multiple areas and ASs, such that an external path computing entity can access the information by passively listening to a route reflector.

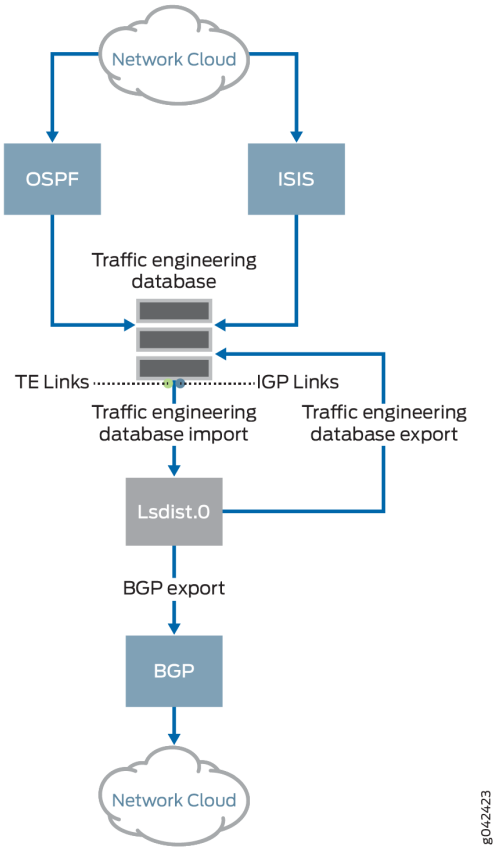
Implementation

In Junos OS, the IGPs install topology information into a database called the traffic engineering database. The traffic engineering database contains the aggregated topology information. To install IGP topology information into traffic engineering database, use the `set igp-topology` configuration statement at the `[edit protocols isis traffic-engineering]` and `[edit protocols ospf traffic-engineering]` hierarchy levels.

The mechanism to distribute link-state information using BGP includes the process of advertising the traffic engineering database into BGP-TE (import), and installing entries from BGP-TE into the traffic engineering database (export).

Starting in Junos OS Release 20.4R1, you can configure IS-IS traffic engineering to store IPv6 information in the traffic engineering database (TED) in addition to IPv4 addresses. BGP-LS distributes this information as routes from the traffic engineering database to the lsdist.0 routing table using the traffic engineering database import policies. These routes are advertised to BGP-TE peers as network layer reachability information (NLRI) with IPv6 router ID type, length, and value (TLV). With addition of IPv6 information, you can benefit from obtaining the complete network topology into the traffic engineering database.

Figure 64: Junos OS Implementation of BGP Link-State Distribution



BGP-LS NLRI and Confederation ID

Starting in Junos OS Release 23.1R1, Junos OS enables BGP Link State (BGP-LS) network layer reachability information (NLRI) to carry the confederation ID in TLV 512 when BGP confederation is

enabled. The NLRI carries the confederation ID along with the member autonomous system number (AS number) in TLV 517 as defined in RFC 9086. The Junos OS traffic engineering database module makes necessary changes to encode confederation ID and member AS number in TLV 512 and TLV 517 respectively, while originating the BGP-LS NLRI (which is injected into lsdist.0 routing table). In releases before Junos OS Release 23.1R1, BGP-LS NLRI carries only the member AS number in TLV 512 and the confederation ID is not encoded in the lsdist.0 routing table.

Traffic Engineering Database Import

To advertise the traffic engineering database into BGP-TE, the link and node entries in the traffic engineering database are converted in the form of routes. These converted routes are then installed by the traffic engineering database on behalf of the corresponding IGP, into a user-visible routing table called `lsdist.0`, on conditions subject to route policies. The procedure of leaking entries from the traffic engineering database into `lsdist.0` is called traffic engineering database import as illustrated in Figure 64 on page 993.

There are polices to govern the traffic engineering database import process. By default, no entries are leaked from the traffic engineering database into the `lsdist.0` table.

Starting in Junos OS Release 17.4R1, the traffic engineering database installs interior gateway protocol (IGP) topology information in addition to RSVP-TE topology information in the lsdist.0 routing table as illustrated in Figure 64 on page 993. Prior to Junos OS Release 17.4R1, the traffic engineering database only exported RSVP-TE topology information. Now you can monitor both IGP and traffic engineering topology information. The BGP-LS reads IGP entries from lsdist.0 and advertises these entries to the BGP peers. To import IGP topology information into BGP-LS from lsdist.0, use the `set bgp-ls` configuration statement at the `[edit protocols mpls traffic-engineering database import igp-topology]` hierarchy level.

Traffic Engineering Database Export

BGP can be configured to export or advertise routes from the `lsdist.0` table, subject to policy. This is common for any kind of route origination in BGP. In order to advertise BGP-TE into the traffic engineering database, BGP needs to be configured with the BGP-TE address family, and an export policy that selects routes for redistribution into BGP.

BGP then propagates these routes like any other NLRI. BGP peers that have the BGP-TE family configured and negotiated receive BGP-TE NLRIs. BGP stores the received BGP-TE NLRIs in the form of routes in the `lsdist.0` table, which is the same table that stores locally originated BGP-TE routes. The BGP-installed routes in `lsdist.0` are then distributed to other peers like any other route. Thus, the standard route selection procedure applies to BGP-TE NLRIs received from multiple speakers.

To achieve interdomain TE, the routes in `lsdist.0` are leaked into the traffic engineering database through a policy. This process is called traffic engineering database export as illustrated in Figure 64 on page 993.

There are polices to govern the traffic engineering database export process. By default, no entries are leaked from the `lsdist.0` table into the traffic engineering database.

Starting in Junos OS Release 22.4R1, you can distribute the traffic engineering (TE) policies that originate from the segment routing protocol to the traffic engineering database (TED) and into the BGP link-state as routes. BGP link-state collects the information related to the TE policies, so that the external controllers can perform actions such as path-computation, re-optimization, and network visualization within and across domains.

Configure `set protocols source-packet-routing traffic-engineering database` to allow the segment routing (SR) policies to be stored in TED.

NOTE: For SDN applications, such as PCE and ALTO, the BGP-TE advertised information cannot leak into the traffic engineering database of a router. In such cases, an external server that peers with the routers using BGP-TE is used to move topology information up into the sky/orchestration system that spans the network. These external servers can be deemed as BGP-TE consumers, where they receive BGP-TE routes, but do not advertise them.

Assigning Credibility Values

Once the entries are installed in the traffic engineering database, the BGP-TE learned information is made available for CSPF path computation. The traffic engineering database uses a protocol preference scheme that is based on credibility values. A protocol with a higher credibility value is preferred over a protocol with a lower credibility value. BGP-TE has the capability to advertise information learned from multiple protocols at the same time, and so in addition to the IGP-installed entries in the traffic engineering database, there can be BGP-TE installed entries that correspond to more than one protocol. The traffic engineering database export component creates a traffic engineering database protocol and credibility level for each protocol that BGP-TE supports. These credibility values are configurable in the CLI.

The credibility order for the BGP-TE protocols is as follows:

- Unknown—80
- OSPF—81
- ISIS Level 1—82
- ISIS Level 2—83
- Static—84
- Direct—85

Cross-Credibility Path Computation

After you assign credibility values, each credibility level is treated as an individual plane. The Constrained Shorted Path First algorithm starts with the highest assigned credibility to the lowest, finding a path within that credibility level.

With BGP-TE, it is essential to compute paths across credibility levels to compute inter-AS paths. For example, different credibility settings are seen on a device from area 0 that computes the path through area 1, because area 0 entries are installed by OSPF, and area 1 entries are installed by BGP-TE.

To enable path computation across credibility levels, include the `cross-credibility-cspf` statement at the `edit protocols mpls`, `[edit protocols mpls label-switched-path` *`lsp-name`*`]`, and `[edit protocols rsvp]` hierarchy levels. At the `[edit protocols rsvp]` hierarchy level, enabling `cross-credibility-cspf` impacts bypass LSPs and loose hop expansion in transit.

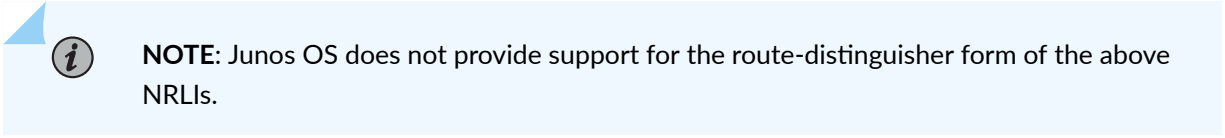
Configuring `cross-credibility-cspf` enables path computation across credibility levels using the Constrained Shortest Path First algorithm, wherein the constraint is not performed on a credibility-by-credibility basis, but as a single constraint ignoring the assigned credibility values.

BGP-TE NLRIs and TLVs

Like other BGP routes, BGP-TE NLRIs can also be distributed through a route reflector that speaks BGP-TE NLRI. Junos OS implements the route reflection support for the BGP-TE family.

The following is a list of supported NLRIs:

- Link NLRI
- Node NLRI
- IPv4 Prefix NLRI (receive and propagate)
- IPv6 Prefix NLRI (receive and propagate)
- TE policy NLRI



NOTE: Junos OS does not provide support for the route-distinguisher form of the above NRLIs.

The following is a list of supported fields in link and node NLRIs:

- Protocol-ID—NLRI originates with the following protocol values:
  - ISIS-L1
  - ISIS-L2

    - OSPF
    - SPRING-TE
- Identifier—This value is configurable. By default, the identifier value is set to `0`.
- Local/Remote node descriptor—These include:
    - Autonomous system
    - BGP-LS Identifier—This value is configurable. By default, the BGP-LS identifier value is set to `0`
    - Area-ID
    - IGP router-ID
- Link descriptors (Only for link NLRI)—This includes:
    - Link Local/Remote Identifiers
    - IPv4 interface address
    - IPv4 neighbor address
    - IPv6 neighbor/interface address—The IPv6 neighbor and interface addresses are not originated, but only stored and propagated when received.
    - Multi-topology ID—This value is not originated, but stored and propagated when received.

The following is a list of supported LINK_STATE attribute TLVs:

- Link attributes:
    - Administrative group
    - Max link bandwidth
    - Max reservable bandwidth
    - Unreserved bandwidth
    - TE default metric
    - SRLG
    - The following TLVs, which are not originated, but only stored and propagated when received:
        - Opaque link attributes
        - MPLS protocol mask

    - Metric
    - Link protection type
    - Link name attribute
- Node attributes:
    - IPv4 Router-ID
    - Node flag bits—Only the overload bit is set.
    - The following TLVs, which are not originated, but only stored and propagated when received:
        - Multi-topology
        - OSPF-specific node properties
        - Opaque node properties
        - Node name
        - IS-IS area identifier
        - IPv6 Router-ID
    - Prefix attributes—These TLVs are stored and propagated like any other unknown TLVs.

Supported and Unsupported Features

Junos OS supports the following features with link-state distribution using BGP:

- Advertisement of multiprotocol assured forwarding capability
- Transmission and reception of node and link-state BGP and BGP-TE NLRIs
- Nonstop active routing for BGP-TE NLRIs
- Policies

Junos OS does **not** support the following functionality for link-state distribution using BGP:

- Aggregated topologies, links, or nodes
- Route distinguisher support for BGP-TE NLRIs
- Multi-topology identifiers
- Multi-instance identifiers (excluding the default instance ID 0)

- Advertisement of the link and node area TLV
- Advertisement of MPLS signaling protocols
- Importing node and link information with overlapping address

BGP Link-State Extensions for Source Packet Routing in Networking (SPRING)

Starting in Junos OS Release 17.2R1, the BGP link-state address family is extended to distribute the source packet routing in networking (SPRING) topology information to software-defined networking (SDN) controllers. BGP typically learns the link-state information from IGP and distributes it to BGP peers. Besides BGP, the SDN controller can get link-state information directly from IGP if the controller is a part of an IGP domain. However, BGP link-state distribution provides a scalable mechanism to export the topology information. BGP link-state extensions for SPRING is supported on interdomain networks.

Source Packet Routing in Networking (SPRING)

SPRING is a control-plane architecture that enables an ingress router to steer a packet through a specific set of nodes and links in the network without relying on the intermediate nodes in the network to decide the actual path it must take. SPRING engages IGPs, such as IS-IS and OSPF, for advertising network segments. Network segments can represent any instruction, topological or service-based. Within IGP topologies, IGP segments are advertised by the link-state routing protocols. There are two types of IGP segments:

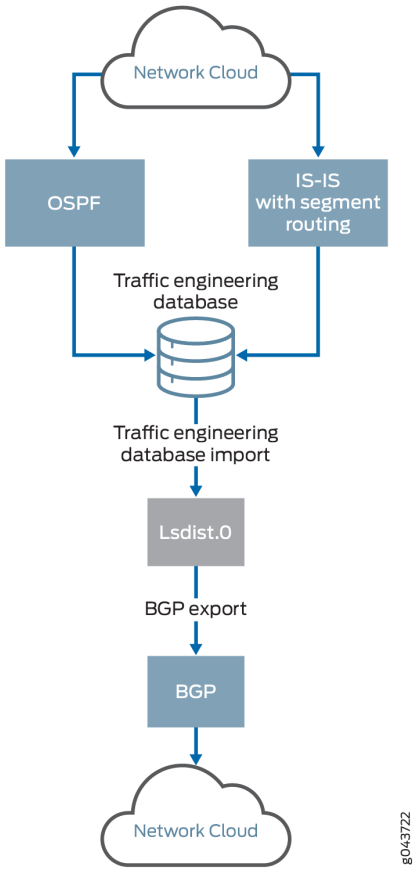
Adjacency segment	A one-hop path over a specific adjacency between two nodes in the IGP
Prefix segment	A multi-hop, equal-cost, multipath-aware shortest-path to a prefix, as per the state of the IGP topology

When SPRING in enabled in a BGP network, BGP link-state address family learns the SPRING information from the IGP link-state routing protocols and advertises segments in the form of segment identifiers (SIDs). BGP link-state address family has been extended to carry SIDs and other SPRING-related information to BGP peers. The route reflector can steer a packet through a desired set of nodes and links by prepending the packet with an appropriate combination of tunnels. This feature allows BGP link-state address family to also advertise the SPRING information to BGP peers.

Flow of BGP Link-State SPRING Data

Figure 65 on page 1000 depicts the data flow of BGP link-state SPRING data that IS-IS pushes to the traffic engineering database.

Figure 65: BGP Link-State Source Packet Routing in Networking (SPRING)



- IGP pushes the SPRING attributes to the traffic engineering database.
- SPRING capabilities and algorithm information are carried forward as node attributes into the traffic engineering database.
- Adjacent SID and LAN adjacent SID information are carried as link attributes.
- Prefix SID or node-SID information is carried as prefix attributes.
- A new set or a change to existing attributes triggers IGP updates to the traffic engineering database with new data.

> ⚠ **CAUTION**: If traffic engineering is disabled at the IGP level, none of the attributes are pushed to the traffic engineering database.

- All parameters in the BGP traffic engineering NLRI, including the link, node, and prefix descriptors are derived from entries in the traffic engineering database.

- The traffic engineering database imports route entries into the `lsdist.0` routing table from IGP subject to policy.
- The default policy of BGP is to export routes, which are known to BGP only. You configure an export policy for non-BGP routes in the `lsdis.0` routing table. This policy advertises an entry learned from the traffic engineering database.

Supported BGP Link-State Attributes and TLVs, and Unsupported Features for BGP Link-State with SPRING

BGP link-state with SPRING supports the following attributes and type, length, and values (TLVs) that are originated, received, and propagated in the network:

Node attributes

- Segment routing Capabilities
- Segment routing Algorithm

Link attributes

- Adjacent-SID
- LAN Adjacent-SID

Prefix descriptors

- IP reachability information

Prefix attributes

- Prefix SID

The following list supports TLVs that are not originated, but only received and propagated in the network:

Prefix descriptors

- Multitopology ID
- OSPF route type

Prefix attributes

- Range
- Binding SID

Junos OS does not support the following features with BGP link-state with SPRING extensions:

- IPv6 prefix origination
- Multitopology identifiers
- Traffic engineering database export for SPRING parameters
- New TLVs with tcpdump (existing TLVs are also not supported).
- SPRING over IPv6

Verifying NLRI Node Learned Through BGP with OSPF as IGP

The following is a sample output to verify the NLRI node learned through BGP with OSPF as the IGP:

Purpose

Verify the lsdist.0 routing table entries.

Action

From operational mode, run the `show route table lsdist.0` command.

```
user@host> show route table lsdist.0 te-node-ip 10.7.7.7 extensive
lsdist.0: 216 destinations, 216 routes (216 active, 0 holddown, 0 hidden)
NODE { AS:65100 Area:0.0.0.1 IPv4:10.7.7.7 OSPF:0 }/1536 (1 entry, 1 announced)
TSI:
LINK-STATE attribute handle 0x61d5da0
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0x61b07cc
                Next-hop reference count: 216
                Source: 10.2.2.2
                Protocol next hop: 10.2.2.2
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State:<Active Int Ext>
                Local AS:   65100 Peer AS:   65100
                Age: 30:22      Metric2: 2
                Validation State: unverified
                Task: BGP_65100.10.2.2.2
                Announcement bits (1): 0-TED Export
                AS path: I
                Accepted
                Area border router: No
```

```
                External router: No
                Attached: No
                Overload: No
                SPRING-Capabilities:
                 - SRGB block [Start: 900000, Range: 90000, Flags: 0x00]
                SPRING-Algorithms:
                 - Algo: 0
                Localpref: 100
                Router ID: 10.2.2.2
                Indirect next hops: 1
                        Protocol next hop: 10.2.2.2 Metric: 2
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.11.1.2 via et-0/0/0.1 weight 0x1
                                Session Id: 0x143
                                10.2.2.2/32 Originating RIB: inet.0
                                  Metric: 2     Node path count: 1
                                  Forwarding nexthops: 1
                                        Nexthop: 10.11.1.2 via et-0/0/0.1
                                        Session Id: 143
```

Meaning

The routes are appearing in the lsdist.0 routing table.

Verifying the Prefix NLRI Learned Through BGP with OSPF as IGP

The following is a sample output to verify the prefix NLRI learned through BGP with OSPF as the IGP:

Purpose

Verify the lsdist.0 routing table entries.

Action

From operational mode, run the `show route table lsdist.0` command.

```
user@host> show route table lsdist.0 te-ipv4-prefix-node-ip 10.7.7.7 extensive
lsdist.0: 216 destinations, 216 routes (216 active, 0 holddown, 0 hidden)
PREFIX { Node { AS:65100 Area:0.0.0.1 IPv4:10.7.7.7 } { IPv4:10.7.7.7/32 } OSPF:0 }/1536 (1
```

```
entry, 0 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0x61b07cc
                Next-hop reference count: 216
                Source: 10.2.2.2
                Protocol next hop: 10.2.2.2
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Active Int Ext>
                Local AS:   65100 Peer AS:   65100
                Age: 30:51      Metric2: 2
                Validation State: unverified
                Task: BGP_65100.10.2.2.2
                AS path: I
                Accepted
                Prefix Flags: 0x00, Prefix SID: 1007, Flags: 0x50, Algo: 0
                Localpref: 65100
                Router ID: 10.2.2.2
                Indirect next hops: 1
                        Protocol next hop: 10.2.2.2 Metric: 2
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.11.1.2 via et-0/0/0.1 weight 0x1
                                Session Id: 0x143
                                10.2.2.2/32 Originating RIB: inet.0
                                  Metric: 2     Node path count: 1
                                  Forwarding nexthops: 1
                                        Nexthop: 10.11.1.2 via et-0/0/0.1
                                        Session Id: 143
```

Meaning

The routes are appearing in the lsdist.0 routing table.

Example: Configuring Link State Distribution Using BGP

IN THIS SECTION

- Requirements | 1005
- Overview | 1006
- Configuration | 1007
- Verification | 1022

This example shows how to configure BGP to carry link-state information across multiple domains, which is used for computing paths for MPLS LSPs spanning multiple domains, such as inter-area TE LSP, and providing a scalable and policy-controlled means for external path computing entities, such as ALTO and PCE, to acquire network topology.

Requirements

This example uses the following hardware and software components:

- Four MX Series routers
- Junos OS Release 14.2 or later running on all the routers

Before you begin:

1. Configure the device interfaces.
2. Configure the autonomous system numbers and router IDs for the devices.
3. Configure the following protocols:
   - RSVP
   - MPLS
   - BGP
   - IS-IS
   - OSPF

Overview

IN THIS SECTION

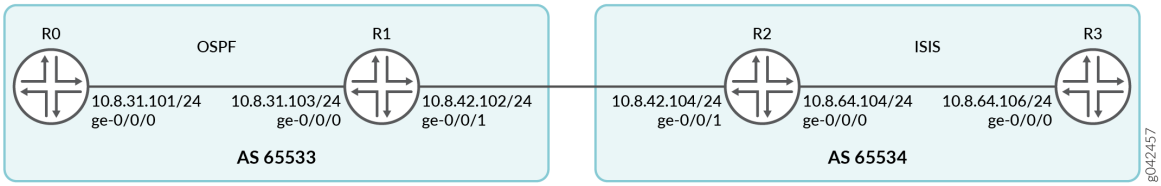
- Topology | **1006**

Starting with Junos OS Release 14.2, a new mechanism to distribute topology information across multiple areas and autonomous systems (ASs) is introduced by extending the BGP protocol to carry link - state information, which was initially acquired using IGP. The IGP protocols have scaling limitations when it comes to distributing large databases. BGP is not only a more scalable vehicle for carrying multi-area and multi-AS topology information, but also provides the policy controls that can be useful for multi-AS topology distribution. The BGP link-state topology information is used for computing paths for MPLS label-switched paths (LSPs) spanning multiple domains, such as inter-area TE LSP, and providing a scalable and policy-controlled means for external path computing entities, such as ALTO and PCE, to acquire network topology.

Starting with Junos OS Release 17.1R1, link state distribution using BGP is supported on QFX10000 switches.

Topology

Figure 66: Link-State Distribution Using BGP



lo0:	
R0	10.22.105.137/32
R1	10.255.105.141/32

lo0:	
R2	10.255.105.139/32
R3	10.255.105.135/32

In Figure 66 on page 1006, Routers R0 and R1 and Routers R2 and R3 belong to different autonomous systems. Routers R0 and R1 run OSPF, and Routers R2 and R3 run IS-IS.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 1007
- Procedure | 1010
- Procedure | 1016

CLI Quick Configuration

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

R0

```
set interfaces ge-0/0/0 unit 0 family inet address 10.8.31.101/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.105.137/32
set routing-options router-id 10.255.105.137
set routing-options autonomous-system 65533
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls traffic-engineering database export policy accept-all
set protocols mpls cross-credibility-cspf
set protocols mpls label-switched-path to-R3-inter-as to 10.255.105.135
set protocols mpls label-switched-path to-R3-inter-as bandwidth 40m
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.105.137
set protocols bgp group ibgp family traffic-engineering unicast
set protocols bgp group ibgp neighbor 10.255.105.141
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
```

```
set policy-options policy-statement accept-all from family traffic-engineering
set policy-options policy-statement accept-all then accept
```

R1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.8.31.103/24
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 family inet address 10.8.42.102/24
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 10.255.105.141/32
set interfaces lo0 unit 0 family iso address 47.0005.0102.5501.8181
set routing-options router-id 10.255.105.141
set routing-options autonomous-system 65533
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.105.141
set protocols bgp group ibgp family traffic-engineering unicast
set protocols bgp group ibgp export nlri2bgp
set protocols bgp group ibgp neighbor 10.255.105.137
set protocols bgp group ebgp type external
set protocols bgp group ebgp family traffic-engineering unicast
set protocols bgp group ebgp neighbor 10.8.42.104 local-address 10.8.42.102
set protocols bgp group ebgp neighbor 10.8.42.104 peer-as 65534
set protocols isis interface ge-0/0/1.0 passive remote-node-iso 0102.5502.4211
set protocols isis interface ge-0/0/1.0 passive remote-node-id 10.8.42.104
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-id
10.8.42.104
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-
router-id 10.255.105.139
set policy-options policy-statement accept-all from family traffic-engineering
set policy-options policy-statement accept-all then accept
set policy-options policy-statement nlri2bgp term 1 from family traffic-engineering
set policy-options policy-statement nlri2bgp term 1 then accept
```

R2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.8.64.104/24
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 family inet address 10.8.42.104/24
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 10.255.105.139/32
set interfaces lo0 unit 0 family iso address 47.0005.0102.5502.4211.00
set routing-options router-id 10.255.105.139
set routing-options autonomous-system 65534
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls traffic-engineering database import policy ted2nlri
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.105.139
set protocols bgp group ibgp family traffic-engineering unicast
set protocols bgp group ibgp export nlri2bgp
set protocols bgp group ibgp neighbor 10.255.105.135
set protocols bgp group ebgp type external
set protocols bgp group ebgp family traffic-engineering unicast
set protocols bgp group ebgp export nlri2bgp
set protocols bgp group ebgp peer-as 65533
set protocols bgp group ebgp neighbor 10.8.42.102
set protocols isis level 1 disable
set protocols isis interface ge-0/0/0.0
set protocols isis interface ge-0/0/1.0 passive remote-node-iso 0102.5501.8181
set protocols isis interface ge-0/0/1.0 passive remote-node-id 10.8.42.102
set protocols isis interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-id
10.8.42.102
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-
router-id 10.255.105.141
set policy-options policy-statement accept-all from family traffic-engineering
set policy-options policy-statement accept-all then accept
set policy-options policy-statement nlri2bgp term 1 from family traffic-engineering
set policy-options policy-statement nlri2bgp term 1 then accept
set policy-options policy-statement ted2nlri term 1 from protocol isis
```

```
set policy-options policy-statement ted2nlri term 1 from protocol ospf
set policy-options policy-statement ted2nlri term 1 then accept
set policy-options policy-statement ted2nlri term 2 then reject
```

R3

```
set interfaces ge-0/0/0 unit 0 family inet address 10.8.64.106/24
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.105.135/32
set interfaces lo0 unit 0 family iso address 47.0005.0102.5502.4250
set routing-options router-id 10.255.105.135
set routing-options autonomous-system 65534
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls traffic-engineering database export policy accept-all
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.105.135
set protocols bgp group ibgp family traffic-engineering unicast
set protocols bgp group ibgp neighbor 10.255.105.139
set protocols isis interface ge-0/0/0.0 level 1 disable
set protocols isis interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set policy-options policy-statement accept-all from family traffic-engineering
set policy-options policy-statement accept-all then accept
```

Procedure

Step-by-Step Procedure

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

To configure Router R1:

1. Configure the Router R1 interfaces.

```
[edit interfaces]
user@R1# set ge-0/0/0 unit 0 family inet address 10.8.31.103/24
user@R1# set ge-0/0/0 unit 0 family iso
user@R1# set ge-0/0/0 unit 0 family mpls
user@R1# set ge-0/0/1 unit 0 family inet address 10.8.42.102/24
user@R1# set ge-0/0/1 unit 0 family iso
user@R1# set ge-0/0/1 unit 0 family mpls
user@R1# set lo0 unit 0 family inet address 10.255.105.141/32
user@R1# set lo0 unit 0 family iso address 47.0005.0102.5501.8181
```

2. Configure the router ID and autonomous system of Router R1.

```
[edit routing-options]
user@R1# set router-id 10.255.105.141
user@R1# set autonomous-system 65533
```

3. Enable RSVP on all the interfaces of Router R1 (excluding the management interface).

```
[edit protocols]
user@R1# set rsvp interface all
user@R1# set rsvp interface fxp0.0 disable
```

4. Enable MPLS on all the interfaces of Router R1 (excluding the management interface).

```
[edit protocols]
user@R1# set mpls interface all
user@R1# set mpls interface fxp0.0 disable
```

5. Configure the BGP group for Router R1 to peer with Router R0, and assign the local address and neighbor address.

```
[edit protocols]
user@R1# set bgp group ibgp type internal
user@R1# set bgp group ibgp local-address 10.255.105.141
user@R1# set bgp group ibgp neighbor 10.255.105.137
```

6. Include the BGP-TE signaling network layer reachability information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@R1# set bgp group ibgp family traffic-engineering unicast
```

7. Enable export of policy nlri2bgp on Router R1.

```
[edit protocols]
user@R1# set bgp group ibgp export nlri2bgp
```

8. Configure the BGP group for Router R1 to peer with Router R2, and assign the local address and neighbor autonomous system to the ebgp BGP group.

```
[edit protocols]
user@R1# set bgp group ebgp type external
user@R1# set bgp group ebgp neighbor 10.8.42.104 local-address 10.8.42.102
user@R1# set bgp group ebgp neighbor 10.8.42.104 peer-as 65534
```

9. Include the BGP-TE signaling NLRI to the ebgp BGP group.

```
[edit protocols]
user@R1# set bgp group ebgp family traffic-engineering unicast
```

10. Enable passive traffic-engineering on the inter-AS link.

```
[edit protocols]
user@R1# set isis interface ge-0/0/1.0 passive remote-node-iso 0102.5502.4211
user@R1# set isis interface ge-0/0/1.0 passive remote-node-id 10.8.42.104
```

11. Enable OSPF on the interface connecting Router R1 to Router R0 and on the loopback interface of Router R1, and enable traffic engineering capabilities.

```
[edit protocols]
user@R1# set ospf traffic-engineering
user@R1# set ospf area 0.0.0.0 interface lo0.0
user@R1# set ospf area 0.0.0.0 interface ge-0/0/0.0
```

12. Enable passive traffic-engineering on the inter-AS link.

```
[edit protocols]
user@R1# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-id 10.8.42.104
user@R1# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-router-id 10.255.105.139
```

13. Configure policies to accept traffic from BGP-TE NLRI.

```
[edit policy-options]
user@R1# set policy-statement accept-all from family traffic-engineering
user@R1# set policy-statement accept-all then accept
user@R1# set policy-statement nlri2bgp term 1 from family traffic-engineering
user@R1# set policy-statement nlri2bgp term 1 then accept
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, 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@R1# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.8.31.103/24;
        }
        family iso;
        family mpls;
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.8.42.102/24;
        }
        family iso;
        family mpls;
```

```
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.105.141/32;
        family iso {
            address 47.0005.0102.5501.8181:00;
        }
    }
}
```

```
user@R1# show routing-options
router-id 10.255.105.141;
autonomous-system 65533;
```

```
user@R1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 10.255.105.141;
        family traffic-engineering {
            unicast;
        }
        export nlri2bgp;
        neighbor 10.255.105.137;
    }
    group ebgp {
```

```
        type external;
        family traffic-engineering {
            unicast;
        }
        neighbor 10.8.42.104 {
            local-address 10.8.42.102;
            peer-as 65534;
        }
    }
}
isis {
    interface ge-0/0/1.0 {
        passive {
            remote-node-iso 0102.5502.4211;
            remote-node-id 10.8.42.104;
        }
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0;
        interface ge-0/0/0.0;
        interface ge-0/0/1.0 {
            passive {
                traffic-engineering {
                    remote-node-id 10.8.42.104;
                    remote-node-router-id 10.255.105.139;
                }
            }
        }
    }
}
```

```
user@R1# show policy-options
policy-statement accept-all {
    from family traffic-engineering;
    then accept;
}
policy-statement nlri2bgp {
    term 1 {
```

```
        from family traffic-engineering;
        then {
            accept;
        }
    }
}
```

Procedure

Step-by-Step Procedure

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

To configure Router R2:

1. Configure the Router R2 interfaces.

```
[edit interfaces]
user@R2# set ge-0/0/0 unit 0 family inet address 10.8.64.104/24
user@R2# set ge-0/0/0 unit 0 family iso
user@R2# set ge-0/0/0 unit 0 family mpls
user@R2# set ge-0/0/1 unit 0 family inet address 10.8.42.104/24
user@R2# set ge-0/0/1 unit 0 family iso
user@R2# set ge-0/0/1 unit 0 family mpls
user@R2# set lo0 unit 0 family inet address 10.255.105.139/32
user@R2# set lo0 unit 0 family iso address 47.0005.0102.5502.4211.00
```

2. Configure the router ID and autonomous system of Router R2.

```
[edit routing-options]
user@R2# set router-id 10.255.105.139
user@R2# set autonomous-system 65534
```

3. Enable RSVP on all the interfaces of Router R2 (excluding the management interface).

```
[edit routing-options]
user@R2# set rsvp interface all
user@R2# set rsvp interface fxp0.0 disable
```

4. Enable MPLS on all the interfaces of Router R2 (excluding the management interface).

```
[edit routing-options]
user@R2# set mpls interface all
user@R2# set mpls interface fxp0.0 disable
```

5. Enable import of traffic engineering database parameters using the ted2nlri policy.

```
[edit protocols]
user@R2# set mpls traffic-engineering database import policy ted2nlri
```

6. Configure the BGP group for Router R2 to peer with Router R3, and assign the local address and neighbor address.

```
[edit protocols]
user@R2# set bgp group ibgp type internal
user@R2# set bgp group ibgp local-address 10.255.105.139
user@R2# set bgp group ibgp neighbor 10.255.105.135
```

7. Include the BGP-TE signaling network layer reachability information (NLRI) to the ibgp BGP group.

```
[edit protocols]
user@R2# set bgp group ibgp family traffic-engineering unicast
```

8. Enable export of policy nlri2bgp on Router R2.

```
[edit protocols]
user@R2# set bgp group ibgp export nlri2bgp
```

9. Configure the BGP group for Router R2 to peer with Router R1.

```
[edit protocols]
user@R2# set bgp group ebgp type external
```

10. Include the BGP-TE signaling NLRI to the ebgp BGP group.

```
[edit protocols]
user@R2# set bgp group ebgp family traffic-engineering unicast
```

11. Assign the local address and neighbor autonomous system to the ebgp BGP group.

```
[edit protocols]
user@R2# set bgp group ebgp peer-as 65533
user@R2# set bgp group ebgp neighbor 10.8.42.102
```

12. Enable export of policy nlri2bgp on Router R2.

```
[edit protocols]
user@R2# set bgp group ebgp export nlri2bgp
```

13. Enable IS-IS on the interface connecting Router R2 with Router R3 and the loopback interface of Router R2.

```
[edit protocols]
user@R2# set isis level 1 disable
user@R2# set isis interface ge-0/0/0.0
user@R2# set isis interface lo0.0
```

14. Enable only IS-IS advertising on the interface connecting Router R2 with Router R1.

```
[edit protocols]
user@R2# set isis interface ge-0/0/1.0 passive remote-node-iso 0102.5501.8181
user@R2# set isis interface ge-0/0/1.0 passive remote-node-id 10.8.42.102
```

15. Configure traffic engineering capability on Router R2.

```
[edit protocols]
user@R2# set ospf traffic-engineering
```

16. Enable only OSPF advertisements on the interface connecting Router R2 with Router R1.

```
[edit protocols]
user@R2# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-id 10.8.42.102
user@R2# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-router-id 10.255.105.141
```

17. Configure policies to accept traffic from the BGP-TE NLRI.

```
[edit policy-options]
user@R2# set policy-statement accept-all from family traffic-engineering
user@R2# set policy-statement accept-all then accept
user@R2# set policy-statement nlri2bgp term 1 from family traffic-engineering
user@R2# set policy-statement nlri2bgp term 1 then accept
user@R2# set policy-statement ted2nlri term 1 from protocol isis
user@R2# set policy-statement ted2nlri term 1 from protocol ospf
user@R2# set policy-statement ted2nlri term 1 then accept
user@R2# set policy-statement ted2nlri term 2 then reject
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, 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@R2# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.8.64.104/24;
        }
        family iso;
        family mpls;
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
```

```
            address 10.8.42.104/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.105.139/32;
        family iso {
            address 47.0005.0102.5502.4211.00;
        }
        family iso;
    }
}
```

```
user@R2# show routing-options
router-id 10.255.105.139;
 autonomous-system 65534;
```

```
user@R2# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    traffic-engineering {
        database {
            import {
                policy ted2nlri;
            }
        }
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
```

```
}
bgp {

    group ibgp {
        type internal;
        local-address 10.255.105.139;
        family traffic-engineering {
            unicast;
        }
        export nlri2bgp;
        neighbor 10.255.105.135;
    }
    group ebgp {
        type external;
        family traffic-engineering {
            unicast;
        }
        export nlri2bgp;
        peer-as 65533;
        neighbor 10.8.42.102;
    }
}
isis {
    level 1 disable;
    interface ge-0/0/0.0;
    interface ge-0/0/1.0 {
        passive {
            remote-node-iso 0102.5501.8181;
            remote-node-id 10.8.42.102;
        }
    }
    interface lo0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/0/1.0 {
            passive {
                traffic-engineering {
                    remote-node-id 10.8.42.102;
                    remote-node-router-id 10.255.105.141;
                }
            }
```

```
        }
    }
}
```

```
user@R2# show policy-options
policy-statement accept-all {
    from family traffic-engineering;
    then accept;
}
policy-statement nlri2bgp {
    term 1 {
        from family traffic-engineering;
        then {
            accept;
        }
    }
}
policy-statement ted2nlri {
    term 1 {
        from protocol [ isis ospf ];
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

Verification

IN THIS SECTION

- Verifying the BGP Summary Status | **1023**
- Verifying the MPLS LSP Status | **1024**
- Verifying the lsdist.0 Routing Table Entries | **1024**
- Verifying the Traffic Engineering Database Entries | **1028**

Verify that the configuration is working properly.

Verifying the BGP Summary Status

Purpose

Verify that BGP is up and running on Routers R0 and R1.

Action

From operational mode, run the `show bgp summary` command.

```
user@R0> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
lsdist.0
                      10         10          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.255.105.141        65533         20         14       0      79        5:18 Establ
  lsdist.0: 10/10/10/0
```

From operational mode, run the `show bgp summary` command.

```
user@R1> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
lsdist.0
                      10         10          0          0          0          0
Peer                    AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.8.42.104           65534         24         17       0      70        6:43 Establ
  lsdist.0: 10/10/10/0
10.255.105.137        65533         15         23       0      79        6:19 Establ
  lsdist.0: 0/0/0/0
```

Meaning

Router R0 is peered with Router R1.

Verifying the MPLS LSP Status

Purpose

Verify the status of the MPLS LSP on Router R0.

Action

From operational mode, run the `show mpls lsp` command.

```
user@R0> show mpls lsp
Ingress LSP: 1 sessions
To              From            State Rt P     ActivePath       LSPname
10.255.105.135  10.255.105.137  Up     0 *                      to-R3-inter-as
Total 1 displayed, Up 1, Down 0

Egress LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Meaning

The MPLS LSP from Router R0 to Router R3 is established.

Verifying the lsdist.0 Routing Table Entries

Purpose

Verify the lsdist.0 routing table entries on Routers R0, R1, and R2.

Action

From operational mode, run the `show route table lsdist.0` command.

```
user@R0> show route table lsdist.0
lsdist.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
NODE { AS:65534 ISO:0102.5502.4211.00 ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
NODE { AS:65534 ISO:0102.5502.4250.00 ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
NODE { AS:65534 ISO:0102.5502.4250.02 ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
NODE { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 OSPF:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:8.42.1.104 } Remote { AS:65534
ISO:0102.5501.8181.00 }.{ IPv4:10.8.42.102 } ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:10.8.64.104 } Remote { AS:65534
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:02:03, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 ISO:0102.5502.4250.00 }.{ IPv4:10.8.64.106 } Remote { AS:65534
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4211.00 }.
{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4250.00 }.
{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
LINK { Local { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 }.{ IPv4:10. 8.42.104 } Remote
{ AS:65534 Area:0.0.0.0 IPv4:10.255.105.141 }.{ IPv4:10.8.42.102 } OSPF:0 }/1152
```

```
                    *[BGP/170] 00:17:32, localpref 100, from 10.255.105.141
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.31.103 via ge-0/0/0.0
```

From operational mode, run the `show route table lsdist.0` command.

```
user@R1> show route table lsdist.0
lsdist.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

NODE { AS:65534 ISO:0102.5502.4211.00 ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
NODE { AS:65534 ISO:0102.5502.4250.00 ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
NODE { AS:65534 ISO:0102.5502.4250.02 ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
NODE { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 OSPF:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:10.8.42.104 } Remote { AS:65534
ISO:0102.5501.8181.00 }.{ IPv4:10.8.42.102 } ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:10.8.64.104 } Remote { AS:65534
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:02:19, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 ISO:0102.5502.4250.00 }.{ IPv4:10.8.64.106 } Remote { AS:65534
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4211.00 }.
```

```
{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4250.00 }.
{ } ISIS-L2:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
LINK { Local { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 }.{ IPv4:10.8.42.104 } Remote
{ AS:65534 Area:0.0.0.0 IPv4:10.255.105.141 }.{ IPv4:10.8.42.102 } OSPF:0 }/1152
                    *[BGP/170] 00:18:00, localpref 100
                       AS path: 65534 I, validation-state: unverified
                     > to 10.8.42.104 via ge-0/0/1.0
```

From operational mode, run the `show route table lsdist.0` command.

```
user@R2> show route table lsdist.0
lsdist.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

NODE { AS:65534 ISO:0102.5502.4211.00 ISIS-L2:0 }/1152
                    *[IS-IS/18] 1d 00:24:39
                       Fictitious
NODE { AS:65534 ISO:0102.5502.4250.00 ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:45
                       Fictitious
NODE { AS:65534 ISO:0102.5502.4250.02 ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:45
                       Fictitious
NODE { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 OSPF:0 }/1152
                    *[OSPF/10] 1d 00:24:39
                       Fictitious
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:10.8.42.104 } Remote { AS:65534
ISO:0102.5501.8181.00 }.{ IPv4:10.8.42.102 } ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:58
                       Fictitious
LINK { Local { AS:65534 ISO:0102.5502.4211.00 }.{ IPv4:10.8.64.104 } Remote { AS:65534
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:02:34
                       Fictitious
LINK { Local { AS:65534 ISO:0102.5502.4250.00 }.{ IPv4:10.8.64.106 } Remote { AS:65534
```

```
ISO:0102.5502.4250.02 }.{ } ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:45
                       Fictitious
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4211.00 }.
{ } ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:45
                       Fictitious
LINK { Local { AS:65534 ISO:0102.5502.4250.02 }.{ } Remote { AS:65534 ISO:0102.5502.4250.00 }.
{ } ISIS-L2:0 }/1152
                    *[IS-IS/18] 00:20:45
                       Fictitious
LINK { Local { AS:65534 Area:0.0.0.0 IPv4:10.255.105.139 }.{ IPv4:10.8.42.104 } Remote
{ AS:65534 Area:0.0.0.0 IPv4:10.255.105.141 }.{ IPv4:10.8.42.102 } OSPF:0 }/1152
                    *[OSPF/10] 00:20:57
                       Fictitious
```

Meaning

The routes are appearing in the lsdist.0 routing table.

Verifying the Traffic Engineering Database Entries

Purpose

Verify the traffic engineering database entries on Router R0.

Action

From operational mode, run the `show ted database` command.

```
user@R0> show ted database
TED database: 5 ISIS nodes 5 INET nodes
ID                           Type Age(s) LnkIn LnkOut Protocol
0102.5501.8168.00(10.255.105.137) Rtr   1046     1      1 OSPF(0.0.0.0)
    To: 10.8.31.101-1, Local: 10.8.31.101, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                           Type Age(s) LnkIn LnkOut Protocol
0102.5501.8181.00            ---    1033     1      0
0102.5502.4211.00(10.255.105.139) Rtr   3519     2      3 Exported ISIS-L2(1)
    To: 0102.5502.4250.02, Local: 10.8.64.104, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
```

```
    To: 0102.5501.8181.00, Local: 10.8.42.104, Remote: 10.8.42.102
      Local interface index: 0, Remote interface index: 0
ID                          Type Age(s) LnkIn LnkOut Protocol
                                                     Exported OSPF(2)
    To: 10.255.105.141, Local: 10.8.42.104, Remote: 10.8.42.102
      Local interface index: 0, Remote interface index: 0
ID                          Type Age(s) LnkIn LnkOut Protocol
0102.5502.4250.00(10.255.105.135) Rtr   1033     1      1 Exported ISIS-L2(1)
    To: 0102.5502.4250.02, Local: 10.8.64.106, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                          Type Age(s) LnkIn LnkOut Protocol
0102.5502.4250.02           Net    1033     2      2 Exported ISIS-L2(1)
    To: 0102.5502.4211.00(10.255.105.139), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: 0102.5502.4250.00(10.255.105.135), 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
10.8.31.101-1                Net    1046     2      2 OSPF(0.0.0.0)
    To: 0102.5501.8168.00(10.255.105.137), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: 10.255.105.141, 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
10.255.105.141              Rtr    1045     2      2 OSPF(0.0.0.0)
    To: 0102.5502.4211.00(10.255.105.139), Local: 10.8.42.102, Remote: 10.8.42.104
      Local interface index: 0, Remote interface index: 0
    To: 10.8.31.101-1, Local: 10.8.31.103, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
```

Meaning

The routes are appearing in the traffic engineering database.

Configuring Link State Distribution Using BGP

You can enable distribution of topology information across multiple areas and autonomous systems (ASs) by extending the BGP protocol to carry link-state information, which was initially acquired using IGP. The IGP protocols have scaling limitations when it comes to distributing large databases. BGP is not only a more scalable vehicle for carrying multi-area and multi-AS topology information, but also provides the policy controls that can be useful for multi-AS topology distribution. The BGP link-state topology

information is used for computing paths for MPLS LSPs spanning multiple domains, such as inter-area TE LSP, and providing a scalable and policy-controlled means for external path computing entities, such as ALTO and PCE, to acquire network topology.

Before you begin:

1. Configure the device interfaces.
2. Configure the router ID and autonomous system number for the device.
3. Configure the following protocols:
   - RSVP
   - MPLS
   - IS-IS
   - OSPF

To enable link-state distribution using BGP:

1. Configure an internal BGP group, and assign the local address and neighbor address for the group.

```
[edit protocols]
user@R1# set bgp group internal-group-name type internal
user@R1# set bgp group internal-group-name local-address ip-address
user@R1# set bgp group internal-group-name neighbor ip-address
```

2. Include the BGP-TE signaling network layer reachability information (NLRI) to the internal BGP group.

```
[edit protocols]
user@R1# set bgp group internal-group-name family traffic-engineering unicast
```

3. Enable export of policy on the device.

```
[edit protocols]
user@R1# set bgp group internal-group-name export second-policy-name
```

4. Configure an external BGP group, and assign the local address and neighbor autonomous system to the group.

```
[edit protocols]
user@R1# set bgp group external-group-name type external
user@R1# set bgp group external-group-name neighbor ip-address local-address ip-address
user@R1# set bgp group external-group-name neighbor ip-address peer-as as-number
```

5. Include the BGP-TE signaling NLRI to the external BGP group.

```
[edit protocols]
user@R1# set bgp group external-group-name family traffic-engineering unicast
```

6. In configuration mode, go to the following hierarchy level:

```
[edit]
user@R1# edit policy-options
```

7. Configure policies to accept traffic from the BGP-TE NLRI.

```
[edit policy-options]
user@R1# set policy-statement policy-name from family traffic-engineering
user@R1# set policy-statement policy-name then accept
user@R1# set policy-statement bgp-import-policy term 1 from family traffic-engineering
user@R1# set policy-statement bgp-import-policy term 1 then next-hop self
user@R1# set policy-statement bgp-import-policy term 1 then accept
```

8. On the remote connecting device, configure policy to accept the OSPF and IS-IS traffic.

```
[edit policy-options]
user@R2# set policy-statement bgp-export-policy term 1 from protocol isis
user@R2# set policy-statement bgp-export-policy term 1 from protocol ospf
user@R2# set policy-statement bgp-export-policy term 1 then accept
user@R2# set policy-statement bgp-export-policy term 2 then reject
```

9. Verify and commit the configuration.

For example:

```
R1
[edit protocols]
user@R1# set rsvp interface all
user@R1# set rsvp interface fxp0.0 disable
user@R1# set mpls interface all
user@R1# set mpls interface fxp0.0 disable
user@R1# set bgp group ibgp type internal
user@R1# set bgp group ibgp local-address 10.255.105.141
user@R1# set bgp group ibgp family traffic-engineering unicast
user@R1# set bgp group ibgp export nlri2bgp
user@R1# set bgp group ibgp neighbor 10.255.105.137
user@R1# set bgp group ebgp type external
user@R1# set bgp group ebgp family traffic-engineering unicast
user@R1# set bgp group ebgp neighbor 8.42.1.104 local-address 8.42.1.102
user@R1# set bgp group ebgp neighbor 8.42.1.104 peer-as 65534
user@R1# set isis interface ge-0/0/1.0 passive remote-node-iso 0102.5502.4211
user@R1# set isis interface ge-0/0/1.0 passive remote-node-id 8.42.1.104
user@R1# set ospf traffic-engineering
user@R1# set ospf area 0.0.0.0 interface lo0.0
user@R1# set ospf area 0.0.0.0 interface ge-0/0/0.0
user@R1# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-
id 8.42.1.104
user@R1# set ospf area 0.0.0.0 interface ge-0/0/1.0 passive traffic-engineering remote-node-
router-id 10.255.105.139
```

```
[edit policy-options]
user@R1# set policy-statement accept-all from family traffic-engineering
user@R1# set policy-statement accept-all then accept
user@R1# set policy-statement nlri2bgp term 1 from family traffic-engineering
```

```
user@R1# set policy-statement nlri2bgp term 1 then next-hop self
user@R1# set policy-statement nlri2bgp term 1 then accept
```

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

```
R2
[edit policy-options]
user@R2# set policy-statement accept-all from family traffic-engineering
user@R2# set policy-statement accept-all then accept
user@R2# set policy-statement nlri2bgp term 1 from family traffic-engineering
user@R2# set policy-statement nlri2bgp term 1 then next-hop self
user@R2# set policy-statement nlri2bgp term 1 then accept
user@R2# set policy-statement ted2nlri term 1 from protocol isis
user@R2# set policy-statement ted2nlri term 1 from protocol ospf
user@R2# set policy-statement ted2nlri term 1 then accept
user@R2# set policy-statement ted2nlri term 2 then reject
```

```
[edit]
user@R2# commit
commit complete
```

Link-State Distribution using SRv6

IN THIS SECTION

- BGP Link-State Extensions for SRv6 | **1034**

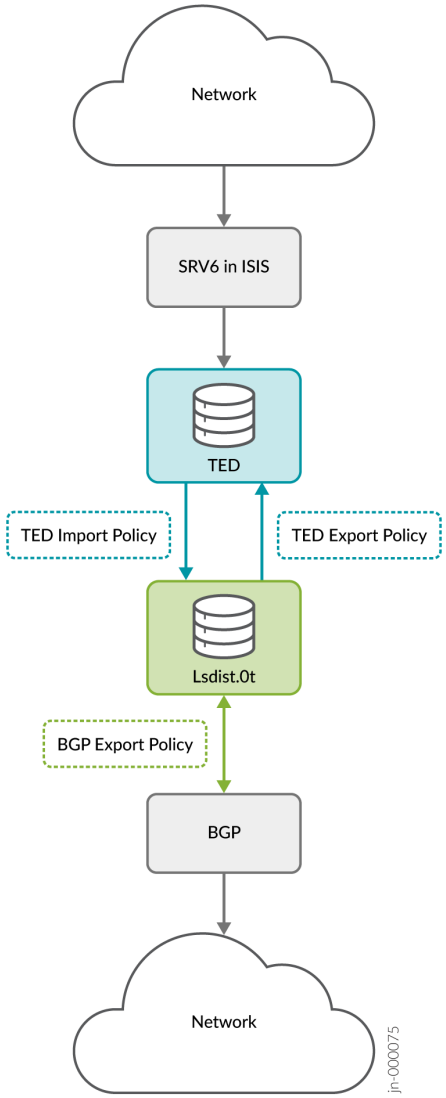
BGP Link-State Extensions for SRv6

Starting in Junos OS Release 21.3R1, we support SRv6 in BGP-LS and Traffic Engineering Database (TED). BGP-LS extensions export the SRv6 topology information to the SDN controllers. Controllers receive the topology information by being part of an IGP domain or through BGP-LS. BGP LS provides a scalable mechanism to export the topology information. It can also be used for the Inter-domain networks. Also, you can now filter NLRI based on IPv6 prefix (SRv6 Locator) and SRv6 SID NLRI.

Flow of BGP Link-State SRv6 Data

BGP LS retrieves the Traffic Engineering (TE) data from the TE Database (TED) and distributes it to the peer BGP Speakers. For this, TED converts its links, nodes and prefixes (IPv4 and IPv6) entries in the form of routes. The following figure shows the data flow in BGP-LS.

Figure 67: BGP Link-State Source Packet Routing in Networking (SRv6)



- SRv6 attributes exchanged via ISIS IGP are now supported in Junos as described in IETF standard [3].
- SRv6 attributes are added into the Traffic Engineering Database (TED).
- SRv6 attributes learned via ISIS IGP are stored in TED as nodes and links are converted to routes. These routes are then subjected to TED import policy and if the policy permits, these are installed in a routing table called lsdist.0.
- BGP can be configured to “export” or advertise routes from lsdist.0 table subject to policy. BGP then propagates these routes like any other NLRI. That is, peers that have BGP-LS family configured and

negotiated receives BGP-LS NLRI's. BGP stores the received BGP-LS NLRIs in the form of routes in "lsdist.0" table, which is the same table that stores locally originated BGP-LS routes. The newly added SRv6 information gets propagated into BGP as attributes of already existing NLRIs (Node, Link and Prefix) and a new SRv6 Locator NLRI.

- The received BGP-LS NLRIs which are installed in the form of routes in "lsdist.0" table can be subjected to TED export policy and if the policy permits, SRv6 attributes from these routes are added into the local instance of TE Database.

IPv6 Prefixes and IPv6 Adjacency SIDs MPLS Support in Traffic Engineering Database and BGP Link-State

IN THIS SECTION

- Benefits of IPv6 Prefixes and IPv6 Adjacency SID MPLS Support in Traffic Engineering Database and BGP-LS | **1037**
- Implementation | **1037**
- Support for Adding the IPv6 Attributes and Information to Traffic Engineering Database from IS-IS | **1038**
- Support for IPv6 Attributes Import from Traffic Engineering Database to lsdist.0 Routing Table | **1038**
- Support for IPv6 Attributes Export to BGP-LS | **1039**
- Support for BGP-LS IPv6 NLRIs and Attributes Export from lsdist.0 Routing Table to Traffic Engineering Database | **1039**
- Configuration Command | **1039**

We have made the following IPv6 enhancements.

- Support for adding the IPv6 attributes and information to traffic engineering database (TED) from Intermediate System to Intermediate System (IS-IS).
- Support for IPv6 attributes import from traffic engineering database to lsdist.0 routing table.
- Support for IPv6 attributes export to BGP Link-State (BGP-LS).
- Support for BGP-LS IPv6 Network Layer Reachability Information (NLRIs) and attributes export from lsdist.0 routing table to traffic engineering database.

NOTE: We support only the IS-IS interior gateway protocol (IGP).

Benefits of IPv6 Prefixes and IPv6 Adjacency SID MPLS Support in Traffic Engineering Database and BGP-LS

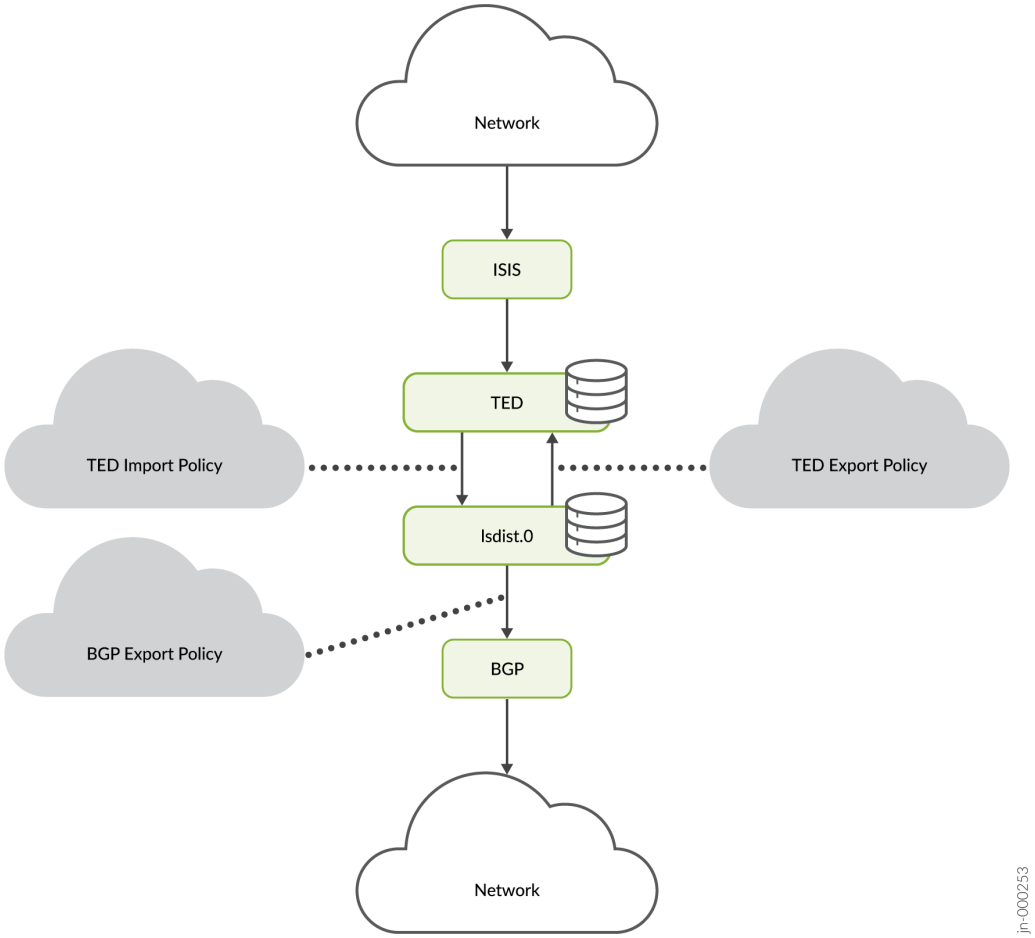
We've enhanced the outputs of the existing operational commands and added the show commands to display the list of IPv6 and IPv4 prefixes, respectively, in the traffic engineering database.

- `show ted database extensive`—Enhanced the output to include the IPv6 segment routing (SR)-MPLS attributes.
- `show ted link detail`—Enhanced the output to include the IPv6 SR-MPLS attributes corresponding to the traffic engineering database links.
- `show route table lsdist.0 [extensive | detail]`—Enhanced the output to include IPv6 NLRIs and IPv6 SR-MPLS attributes.
- `show route`—Included additional parameters to filter entries for viewing in the lsdist.0 table. We've added additional options to include IPv6 prefixes. The options are `te-ipv6-prefix-ipv6-addr` and `te-ipv6-prefix-node-iso`.
- `show ted ipv6-prefix`—Added the show command to display the list of IPv6 prefixes in traffic engineering database.
- `show ted ipv4-prefix`—Added the show command to display the list of IPv4 prefixes in traffic engineering database.

Implementation

BGP-LS retrieves the Traffic Engineering (TE) data from the traffic engineering database and distributes the data to its BGP peers. To achieve this, traffic engineering database converts its links, nodes, and prefix (IPv4 and IPv6) entries in the form of routes. The following figure depicts the flow of information from BGP-LS and towards BGP-LS.

Figure 68: Flow of Information from BGP-LS and towards BGP-LS



Support for Adding the IPv6 Attributes and Information to Traffic Engineering Database from IS-IS

Junos OS supports SR-MPLS attributes for IPv6 data plane, exchanged through IS-IS IGP. As a result of this enhancement, IPv6 attributes and information can be added to the Traffic Engineering Database (TED).

Support for IPv6 Attributes Import from Traffic Engineering Database to lsdist.0 Routing Table

IPv6 attributes received from IS-IS IGP and stored in traffic engineering database as nodes, links, and prefixes are converted to routes. These routes are then subjected to the traffic engineering database import policy. If the policy permits, the routes are installed in a routing table called lsdist.0.

Support for IPv6 Attributes Export to BGP-LS

BGP is configured to export or advertise routes from lsdist.0 table, subject to the policy. It is a routine scenario for any route origination in BGP. BGP then propagates these routes like any other NLRI to the peers with BGP-LS configured and established BGP neighborship. BGP stores the received BGP-LS NLRIs in the form of routes in the lsdist.0 table, which is the same table that stores locally originated BGP-LS routes. As a result of this functionality, newly added IPv6 information is propagated to BGP as attributes of already existing Link NLRI, and as a new IPv6 Prefix NLRI.

Support for BGP-LS IPv6 NLRIs and Attributes Export from lsdist.0 Routing Table to Traffic Engineering Database

In Junos OS, the received BGP-LS NLRIs installed in the form of routes in the lsdist.0 table are subjected to the traffic engineering database export policy. If the policy permits, IPv6 attributes, and information from these routes are then added to the local instance of the traffic engineering database.

Configuration Command

BGP-TE policy command is enhanced to allow filtering of NLRIs based on IPv6 prefix NLRI. See *ipv6-prefix*.

SEE ALSO

- *show ted database*
- *show ted link*
- *show-ted-ipv6-prefix*
- *show-ted-ipv4-prefix*
- *show route*
- *show route table*
- *ipv6-prefix*

Change History Table

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

Release	Description
23.1R1	Starting in Junos OS Release 23.1R1, Junos OS enables BGP Link State BGP-LS NLRI to carry the confederation ID in TLV 512 when BGP confederation is enabled. The NLRI carries the confederation ID along with the member AS number in TLV 517 as defined in RFC 9086.

22.1R1	Starting in Junos OS Release 22.1 R1, we have added IPv6 prefixes and IPv6 adjacency SID MPLS support in the traffic engineering database (TED) and BGP Link-State (LS).
20.4R1	Starting in Junos OS Release 20.4R1, you can configure IS-IS traffic engineering to store IPv6 information in the traffic engineering database (TED) in addition to IPv4 addresses.
17.4R1	Starting in Junos OS Release 17.4R1, the traffic engineering database installs interior gateway protocol (IGP) topology information in addition to RSVP-TE topology information in the lsdist.0 routing table
17.2R1	Starting in Junos OS Release 17.2R1, the BGP link-state address family is extended to distribute the source packet routing in networking (SPRING) topology information to software-defined networking (SDN) controllers.
17.1R1	Starting with Junos OS Release 17.1R1, link state distribution using BGP is supported on QFX10000 switches.

6

CHAPTER

Configuring Graceful Restart for BGP

IN THIS CHAPTER

- Understanding Graceful Restart for BGP | **1042**

Understanding Graceful Restart for BGP

IN THIS SECTION

- Understanding the Long-Lived BGP Graceful Restart Capability | **1042**
- Understanding Maximum Period Configuration for Automatic Generation of BGP Keepalives by Kernel Timers After Switchover | **1044**
- Interoperation of Functionalities With BGP Long-Lived Graceful Restart | **1045**
- Monitoring and Administering BGP Long-Lived Graceful Restart | **1047**
- Increasing the Duration for Preserving BGP Routes Across Slowly-Restarting Peers By BGP Long-Lived Graceful Restart | **1050**
- Configuring BGP Long-Lived Graceful Restart Communities in Routing Policies | **1054**
- Configuring Long-Lived Graceful Restarter Mode Negotiation for a Specific Address Family in Logical Systems and Routing Instances | **1057**
- Informing the BGP Helper Router or Peer About Retaining Routes By Configuring the Forwarding State Bit for All Address Families and for a Specific Address Family | **1062**
- Example: Preserving Route Details for Slow and Latent BGP Peers By Using BGP Long-Lived Graceful Restart | **1068**

Understanding the Long-Lived BGP Graceful Restart Capability

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

Historically, routing protocols and BGP, in particular, have been designed with a focus on correctness, where a significant aspect of the "correctness" is for each network element's forwarding state to converge toward the current state of the network as quickly as possible. For this reason, the protocol was designed to remove state advertised by routers which went down (from a BGP perspective) as promptly as possible. Using BGP Graceful Restart defined in RFC 4724, the fast convergence functionality has been an attempt to rapidly remove "stale" state from the network.

Over a period of time, two contributing factors have caused this method of rapid removal of stale states to be modified and enhanced. The first is the widespread adoption of tunneled forwarding infrastructures, for example MPLS. Such infrastructures eliminate the risk of some types of forwarding loops that can arise in hop-by-hop forwarding, and thereby reduce one of the motivations for strong

consistency between forwarding elements. The second is the increasing use of BGP as a transport for data less closely associated with packet forwarding than was originally the case. Examples include the use of BGP for autodiscovery (VPLS [RFC4761]) and filter programming (FLOWSPEC [RFC5575]). In these cases, BGP data assumes a characteristic that is not in line with traditional routing.

It was important to offer network operators the ability to choose to retain BGP data for a longer period when the BGP control plane fails for some reason. Although the properties of BGP Graceful Restart are close to this desired requirement to preserve BGP information for a longer duration, several gaps exist, most notably in maximum time for which "stale" information can be retained—graceful restart imposes a 4095-second upper-bound limitation. Junos OS supports a BGP capability called long-lived graceful restart capability so that stale information can be retained for a longer time across a session reset. It also supports a new BGP community, "LLGR_STALE", to mark such information. Such stale information is to be treated as least-preferred, and its advertisement limited to BGP speakers that support the new capability.

BGP long-lived graceful restart (LLGR) allows a network operator to choose to maintain stale routing information from a failed BGP peer much longer than the existing BGP Graceful Restart facility. This functionality to maintain the BGP routes for a longer time period is in accordance with the IETF draft, *Support for Long-lived BGP Graceful Restart—draft-uttaro-idr-bgp-persistence-03*. According to this draft, long-lived graceful restart (LLGR) must be explicitly configured per NLRI, and it includes provisions to prevent the spread of stale information to other peers that do not recognize and validate LLGR. The following benefits and operations are caused by LLGR:

- Routes from failed nodes are retained for a configured time period (on the order of days).
- You can examine per-NLRI LLGR negotiation states using appropriate show commands.
- You can view whether LLGR is currently in effect for a peer, and if it is effective, the period after which it expires.
- Stale routes retained by LLGR are explicitly marked in the output of the `show bgp neighbor` command.
- Stale routes learned from other neighbors are explicitly marked in the output of the `show bgp neighbor` command (using well-defined communities).

Although the LLGR methodology can be applied to a number of different scenarios, one specific scenario is the salient objective of this feature. In a scenario in which a loss of connectivity between a route reflector and a client occurs, including intermittent connectivity which can cause a connection to be reset before the entire RIB can be transmitted, such a failure does not result in a restart. Also, such a phenomenon does not imply that any sort of connectivity problem between the clients and the next-hops advertised by the route reflector exists. It is anticipated that a typical long-lived restart time is in the order of 12 hours.

All of the behavioral guidelines and operational points described in the IETF draft, *draft-uttaro-idr-bgp-persistence-03*, for LLGR are supported. Also, backward compatibility with existing Junos OS features in releases earlier than Release 15.1, specifically graceful restart and nonstop routing (NSR), is supported.

When LLGR is configured, graceful restart operates in the existing manner, except as explicitly illustrated in the Internet draft. You can also configure both LLGR and NSR at the same time, and achieve the complete LLGR functionality. As a prerequisite for LLGR, support for the IETF draft, *Notification Message support for BGP Graceful Restart—draft-ietf- idr-bgp-gr-notification-01*, is implemented. This draft extends the behavior of ordinary GR to allow it to protect against communications interruptions and protocol errors.

SEE ALSO

Monitoring and Administering BGP Long-Lived Graceful Restart | **1047**

Understanding Maximum Period Configuration for Automatic Generation of BGP Keepalives by Kernel Timers After Switchover

In Junos OS, nonstop active routing (NSR) uses the same infrastructure as graceful Routing Engine switchover (GRES) to preserve interface and kernel information. However, NSR also saves routing protocol information by running the routing protocol process (rpd) on the backup Routing Engine. By saving this additional information, NSR is self-contained and does not rely on helper routers (or switches) to assist the routing platform in restoring routing protocol information. NSR is advantageous in networks where neighbor routers (or switches) do not support graceful restart protocol extensions. As a result of this enhanced functionality, NSR is a natural replacement for graceful restart.

Nonstop active routing automerge is one of the kernel components of the socket replication. On switchover, this component merges the socket pairs automatically from the backup to the primary Routing Engine. NSR switchover from backup to primary happens when rpd issues a merge call for each secondary socket pair to merge them to a single socket, which could result in a delay. To avoid this delay, an automerge module in the kernel decouples the secondary socket merge from rpd and automatically merges secondary sockets on switchover so that the rpd high priority thread takes advantage of this and generates faster keepalive to sustain TCP connections on switchover.

By default, BGP does not register for the automatic keepalive generation service provided by the kernel right after the switchover event from backup to primary. For this, you need to enable the `nonstop-routing-options` statement at [`edit routing-options`] hierarchy level and configure precision timers in BGP. Configuring precision timers in BGP allows BGP to register all of its sessions with the automatic keepalive generation service provided by the kernel. Once registered, the kernel automatically generates keepalives using its timers on behalf of BGP for its control sessions just after the switchover event from backup to primary. This allows generation of more reliable keepalives for control sessions with very small timers during the switchover event.

SEE ALSO

nonstop-routing-options

Interoperation of Functionalities With BGP Long-Lived Graceful Restart

IN THIS SECTION

- Limitations on Supported NLRIs | **1045**
- LLGR Restarter Mode Under NSR | **1046**
- LLGR Capability At Global, BGP Group, and BGP Neighbor Levels | **1046**

This topic contains the following sections that describe the working behavior of different functionalities with BGP long-lived graceful restart and the various system conditions:

Starting in Junos OS Release 15.1, Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

Limitations on Supported NLRIs

LLGR configuration and capability negotiation is supported for the following BGP network layer reachability information (NLRI) families:

- l2vpn
- inet labeled-unicast
- inet flow
- route-target
- inet-vpn unicast
- inet-vpn flow
- inet6-vpn unicast

LLGR configuration and capability negotiation is prevented for the following families:

- inet-mvpn
- inet6-mvpn
- inet-mdt

For the NLRI families for which LLGR capability is prevented, it indicates that an attempt to commit a configuration that includes an LLGR configuration for these families is rejected, and such settings are not saved. The NLRIs associated with these families are not included in an LLGR capabilities advertisement, and are disregarded in a received LLGR capabilities advertisement.

LLGR configuration and capability negotiation is permitted, but hidden, for other families.

LLGR Restarter Mode Under NSR

When NSR and LLGR are configured together, the router negotiates the LLGR capability in the usual, regular manner, including a long-lived stale time to trigger LLGR receiver mode in its peers. However, full LLGR restarter functionality (delaying the transmission of End of RIB markers until EoRs are received from all peers) does not function under NSR. During a full system (both Routing Engines) restart, the routing protocol daemon (rpd) does not wait for EoRs from other peers before sending its own EoR. It transmits the EoR as soon as it has transmitted the current RIB contents. This condition can cause transient outages when the network reconverges. NSR is considered to be adequate to handle all single-Routing Engine restart scenarios. The restarter mode restriction effects only scenarios where both Routing Engines (or both copies of rpd) restart simultaneously. Ordinary restarter mode configuration is not enabled with NSR.

Ordinary graceful-restart restarter mode configuration continues to be not supported with NSR.

LLGR Capability At Global, BGP Group, and BGP Neighbor Levels

Long-lived graceful restart receiver mode is enabled by default, unless ordinary graceful restart receiver mode is disabled. To enable the BGP long-lived graceful restart (LLGR) capability, include the `long-lived receiver enable` statement at the `[edit protocols bgp graceful-restart]` hierarchy level. Apart from enabling BGP LLGR at the global or system-wide level, you can also include the long-lived receiver enable statement at the `[edit protocols bgp group` *`group-name`* `graceful- restart]` hierarchy level to configure LLGR for a particular BGP group and at the `[edit protocols bgp group` *`group-name`* `neighbor` *`neighbor-address`* `graceful-restart]` hierarchy level to configure LLGR for a particular BGP neighbor. To disable the BGP LLGR mechanism, include the `long-lived receiver disable` option the `[edit protocols bgp graceful-restart]`, `[edit protocols bgp group` *`group-name`* `graceful-restart]`, or [edit protocols bgp group-group-name neighbor neighbor-address graceful-restart] hierarchy level. Disabling LLGR deactivates all of the LLGR capabilities (both receiver and restarter modes) for all NLRI families. This property is inherited by groups from the global configuration, and by neighbors from the group configuration.

SEE ALSO

Understanding Maximum Period Configuration for Automatic Generation of BGP Keepalives by Kernel Timers After Switchover | 1044

Monitoring and Administering BGP Long-Lived Graceful Restart

This topic describes the operational commands and their significance to enable you analyze and view the parameters related to BGP long-lived graceful restart. You can analyze the statistical counters and metrics related to any traffic loss and take an appropriate corrective measure. The fields displayed in the output of the show commands help in diagnosing and debugging network performance and traffic-handling efficiency problems.

The `clear bgp neighbor` *`neighbor-address`* `stale-routes` causes any stale routes currently being held for the specified neighbor because of graceful restart (GR) or long-lived graceful restart (LLGR) receiver mode operations. The `clear bgp neighbor` *`neighbor-address`* `gracefully` command is the same as `clear bgp neighbor hard` (the default for `clear bgp neighbor`), but it does not use the new Hard Reset subcode on the Notify and Cease messages that are sent. This allows the neighbor to enter GR or LLGR helper mode, if negotiated. The session is still cleared on this router, and this router does not enter GR or LLGR helper mode.

A hidden `clear` command is available added for the BGP long-lived graceful restart capability for debugging purposes:

`clear bgp neighbor` *`neighbor-address`* `socket`.

This command breaks the TCP connection for an established peering session. This is the only direct implication of the command and all other implications are side effects of the connection being broken. The resultant effect is that (unless GR notification extensions have been disabled) both sides of the connection will enter GR or LLGR helper mode, if negotiated, and the TCP connection will be reestablished.

The output of the `show bgp neighbor` command is enhanced to display the following additional information:

- The long-lived graceful restart option
- The LLGR parameters that the peer has negotiated
- The LLGR parameters that the restart router has negotiated
- Times are displayed using the routing protocol daemon (rpd) %#0T format:

  `<weeks>w<days>d <hours>:<minutes>:<seconds>`

  Zero leading elements are omitted, for example, a value less than one week do not include the weeks.

If long-lived graceful restart is completely disabled for a neighbor, the following is displayed:

```
user@router> show bgp neighbor
  Peer: 10.6.128.225+45824 AS 100 Local: 10.255.255.14+44542 AS 100
    Type: Internal    State: Established    Flags: <Sync>
    Last State: OpenConfirm   Last Event: RecvKeepAlive
    Last Error: None
    Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
    Options: <LLGRHelperDisabled> {The LLGRHelperDisabled value for the Options field denotes
that long-lived BGP graceful restart is completely disabled for a neighbor}
```

If a neighbor does not support LLGR entirely, the following is displayed:

```
user@router> show bgp neighbor
  ...
    Peer does not support LLGR Restarter or Receiver functionality {BGP neighor or peer does not
support long-lived BGP graceful restart restarter or receiver
functionality}
```

While LLGR receiver mode is active (a peer that negotiated LLGR has disconnected and not yet reconnected), the output of the `show bgp neighbor` command displays the amount of time left until the LLGR expires, the time remaining on the GR stale timer, and RIB details:

```
  user@router> show bgp neighbor
  Peer: 10.4.12.11 AS 100        Local: 10.6.128.225 AS 100
    Type: Internal    State: Active         Flags: <>
    Last State: Idle          Last Event: Start
    Last Error: None
    Export: [ foo ]
    Options: <Preference LocalAddress Refresh GracefulRestart>
    Options: <LLGR>
    Local Address: 10.6.128.225 Holdtime: 90 Preference: 170
    Number of flaps: 3
    Last flap event: Restart
    Error: 'Cease' Sent: 0 Recv: 1
    Time until long-lived stale routes deleted: inet-vpn-unicast 10:00:22 route-target 10:00:22
    Table bgp.l3vpn.0
      RIB State: BGP restart is complete
      RIB State: VPN restart is complete
      Send state: not advertising
      Active prefixes:              0
```

```
    Received prefixes:            7
    Accepted prefixes:            7
    Suppressed due to damping:    0
  Table foo.inet.0 Bit: 30000
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not in sync
    Active prefixes:              0
    Received prefixes:            7
    Accepted prefixes:            7
    Suppressed due to damping:    0
```

When BGP graceful restart receiver mode is active for a neighbor, additional information is displayed in the output of the `show bgp neighbor` command. These details include the list of NLRIs that stale routes are held for (NLRI we are holding stale routes for field), the time remaining on the restart timer (Time until stale routes are deleted or become long-lived stale field), the time remaining on the stale timer (Time until end-of-rib is assumed for stale routes), and the RIB details. Time is displayed in Coordinated Universal Time (UTC) format (YYYY-MM-DD-HH:MM:SS). Note that the stale timer display ('Time until end-of-rib is assumed') is also present when a session is active, but the neighbor as not yet sent all of the end-of-rib indications.

When graceful restart or LLGR helper mode is active, the RIB information is now displayed by the `show bgp summary` command. If a BGP session is established on the main routing device, the field shows the number of active, received, accepted, and damped routes that are received from a neighbor and appear in the inet.0 (main) and inet.2 (multicast) routing tables. For example, 8/10/10/2 and 2/4/4/0 indicate the following:

- 8 active routes, 10 received routes, 10 accepted routes, and 2 damped routes from a BGP peer appear in the inet.0 routing table.
- 2 active routes, 4 received routes, 4 accepted routes, and no damped routes from a BGP peer appear in the inet.2 routing table.

The `show route detail` command (with and without the `receive-protocol bgp` option) is enhanced to identify routes that are held in the long-lived stale state. The `LongLivedStale` flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. The `LongLivedStaleImport` flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. One or both of these flags may be displayed for a route. Neither of these flags will be displayed at the same time as the Stale (ordinary GR stale) flag. When a route is de-preferenced because it is long-lived stale, the Inactive reason field in the output of the show route detail command

displays LLGR stale. The new LLGR stale inactive reason fits into the route selection hierarchy between Preference and Local preference.

```
user@router> show route receive-protocol bgp 10.4.12.11 detail

  bgp.l2vpn.0: 38 destinations, 39 routes (37 active, 0 holddown, 1 hidden)
  * 1.1.1.4:100:1.1.1.4/96 AD (1 entry, 1 announced)
    Accepted LongLivedStale LongLivedStaleImport
    Nexthop: 10.4.12.11
    Localpref: 100
    AS path: I
```

TIP: According to the Juniper Technical Assistance Center (JTAC), one helpful command to help troubleshoot issues related to BGP long-lived graceful restart is the `show route table bgp.l2vpn.0 detail hidden` command. The output of the command helps you detect if the BGP routes still exist after the BGP session has ended. Use of the `hidden` option enables you to see the routes during and after an incident, and discover information that explains why the routes are hidden. Other clues that help you troubleshoot this scenario include the appearance of stale BGP log entries (such as `bgp_mark_route_stale`), and hidden routes showing up in the output of the `show bgp summary` command.

SEE ALSO

Interoperation of Functionalities With BGP Long-Lived Graceful Restart | **1045**

Increasing the Duration for Preserving BGP Routes Across Slowly-Restarting Peers By BGP Long-Lived Graceful Restart

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

Long-lived graceful restart receiver mode is enabled by default, unless ordinary graceful restart receiver mode is disabled. To enable the BGP long-lived graceful restart (LLGR) capability, include the `long-lived receiver enable` statement at the `[edit protocols bgp graceful-restart]` hierarchy level. Apart from enabling BGP LLGR at the global or system-wide level, you can also include the long-lived receiver enable statement at the `[edit protocols bgp group` *`group-name`* `graceful-restart]` hierarchy level to configure LLGR for

a particular BGP group and at the [edit protocols bgp group *group-name* neighbor *neighbor-address* graceful-restart] hierarchy level to configure LLGR for a particular BGP neighbor. To disable the BGP LLGR mechanism, include the long-lived receiver disable option the [edit protocols bgp graceful-restart], [edit protocols bgp group *group-name* graceful-restart], or [edit protocols bgp group-group-name neighbor neighbor-address graceful-restart] hierarchy level. Disabling LLGR deactivates all of the LLGR capabilities (both receiver and restarter modes) for all NLRI families. This property is inherited by groups from the global configuration, and by neighbors from the group configuration.

BGP neighbors can be configured at the following hierarchy levels:

- [edit protocols bgp group *group-name*]—Default logical system and default routing instance.
- [edit routing-instances *instance-name* protocols bgp group *group-name*]—Default logical system with a specified routing instance.
- [edit logical-systems *logical-system-name* protocols bgp group *group-name*]—Configured logical system and default routing instance.
- [edit logical-systems *logical-system-name* routing-instances *instance-name* protocols bgp group *group-name*]—Configured logical system with a specified routing instance.

The long-lived receiver enable overrides a disable option inherited from a higher level in the configuration. It does not enable long-lived graceful restart restarter mode for all families—restarter mode must be configured explicitly for each family.

To enable LLGR-stale routes to be advertised to neighbors that do not advertise the LLGR capability, include the advertise-to-non-llgr-neighbor statement at the [edit protocols bgp graceful-restart long-lived], [edit protocols bgp group *group-name* graceful-restart long-lived], or [edit protocols bgp group *group-name* neighbor *neighbor-address* graceful-restart long-lived] hierarchy level. This setting applies to both routes that were marked LLGR-stale by this router, and LLGR-stale routes received from neighbors. Ideally, all routers in an autonomous system support the IETF draft specification before it was enabled. However, to facilitate incremental deployment, stale routes might be required to be advertised to neighbors that have not advertised the long-lived graceful restart capability under the following conditions: The neighbors must be internal (IBGP or Confederation) neighbors. The NO_EXPORT community must be attached to the stale routes. The stale routes must have their LOCAL_PREF attribute set to zero. If this technique for partial deployment is used, you must set LOCAL_PREF to zero for all LLGR routes throughout the autonomous system. This configuration trades off a small reduction in flexibility (ordering may not be preserved between competing LLGR routes) for consistency between routers that support and do not support this specification. Because consistency of route selection can be important for preventing forwarding loops, the latter consideration of routers that do not support this specification precedes.

To avoid the no-export BGP community from being automatically added to routes advertised to external BGP neighbors (presumed to be CE routers), include the omit- no-export statement at the [edit protocols bgp graceful-restart long-lived], [edit protocols bgp group *group-name* graceful-restart long-lived], or [edit protocols bgp group *group-name* neighbor *neighbor-address* graceful-restart long-lived] hierarchy level. In VPN

deployments, for example, BGP is often used as a PE-CE protocol. It might be a practical necessity in such deployments to accommodate interoperation with CEs that cannot easily be upgraded to support specifications such as this one. This requirement causes a problem while ensuring that "stale" routing information does not leak beyond the perimeter of routers that support these procedures where one or more IBGP routers are not upgraded. In the VPN PE-CE case, the protocol in use is EBGP, and the LOCAL_PREF, an IBGP-only path attribute, is used. The principal motivation for restricting the propagation of "stale" routing information is the reason to prevent it from spreading without limit once it exits the BGP confederation boundary. VPN deployments are typically topologically constrained, removing this concern. For this reason, an implementation might advertise stale routes over a PE-CE session, when explicitly configured. In such a scenario, the implementation must attach the NO_EXPORT community to the routes in question by default, as an additional protection against stale routes spreading without limit. Attachment of the NO_EXPORT community can be disabled explicitly to accommodate exceptional cases. It might be necessary to advertise stale routes to a CE in some VPN deployments, even if the CE does not support this specification. In that case, if you configure the PE routers to advertise such routes, you must notify the operator of the CE receiving the routes, and the CE must be configured to depreference the routes. Typical BGP implementations perform this operation by matching on the LLGR_STALE community, and setting the LOCAL_PREF for matching routes to zero.

When the LLGR receiver mode is enabled or disabled, the session is reset. This behavior enables the new capability value to be sent to the neighbor. When the `advertise-to-non-llgr-neighbor` option is enabled or disabled, export policy is reevaluated, and LLGR stale routes might be advertised or withdrawn. When the `omit-no-export` option is added or removed, the session is reset. This rest of a session enables LLGR stale routes to be readvertised with or without the no- export community (which is added outside of the export policy).

To enable the BGP long-lived graceful restart capability at the system or global level and configure its properties:

```
[edit]
protocols {
    bgp {
        graceful-restart {
            long-lived {
                receiver {
                    enable:
                    disable;
                }
                advertise-to-non-llgr-neighbor {
                    omit-no-export;
                }
            }
        }
```

```
    }
}
```

To enable the BGP long-lived graceful restart capability at the BGP group level and configure its properties:

```
[edit]
protocols {
    bgp {
        group group-name {
            graceful-restart {
                long-lived {
                    receiver {
                        enable:
                        disable;
                    }
                    advertise-to-non-llgr-neighbor {
                        omit-no-export;
                    }
                }
            }
        }
    }
}
```

To enable the BGP long-lived graceful restart capability at the neighbor or peer group level and configure its properties:

```
[edit]
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    long-lived {
                        receiver {
                            enable:
                            disable;
                        }
                        advertise-to-non-llgr-neighbor {
                            omit-no-export;
```

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

SEE ALSO

Monitoring and Administering BGP Long-Lived Graceful Restart | 1047

Configuring BGP Long-Lived Graceful Restart Communities in Routing Policies

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

Two new well-known communities are introduced. These new BGP communities can be used in any of the configuration hierarchy levels as other symbolic well-known communities (such as no-advertise, no-export, and no-export-subconfed) in the community attribute of static route definitions or in a policy-options community definition. The two new communities are as follows:

- `llgr-stale`—Adds a community to a long-lived stale route when it is readvertised.
- `no-llgr`—Marks routes which a BGP speaker does not want to be retained by LLGR. The Notification message feature does not have any associated configuration parameters.

You can include the `llgr-stale` and `no-llgr` options with the `community` *`name`* `members` statement to associate BGP community information with a static, aggregate, or generated route at the following hierarchy levels:

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

To configure the BGP long-lived graceful restart communities for use in a routing policy match condition:

```
[edit policy-options]
community name {
    members [ llgr-stale | nollgr];
}
```

Configuring LLGR does not require that BGP graceful restart also be configured. The values for the llgr-stale and no-llgr well-known communities are 0xFFFF0006 and 0xFFFF0007 respectively. The privileges are the same as for protocols bgp. The long-lived-graceful-restart section is visible only for families l2vpn, inet labeled-unicast, inet flow and route-target. It is prohibited for inet-mvpn, inet6-mvpn and inet-mdt. It is hidden for other families.

Junos OS also provides support for configuring a BGP export policy that matches the state of a route for BGP long-lived graceful restart. You can associate the community that you previously defined and a list of address prefixes in a routing policy to selectively accept or reject the routes for long-lived graceful restart for the specified prefixes, as follows:

```
policy-options {
    prefix-list name;
    community name members [ llgr-stale | nollgr];
    policy-statement name{
        from {
            prefix-list name;
            community name;
        }
        then {
            (accept | reject)
        }
    }
}
```

Two hidden configuration statements are added under the [edit protocols bgp graceful-restart] hierarchy level for global, group-level, and neighbor group-level configuration.

The disable-notification-flag statement at the [edit protocols bgp graceful-restart], [edit protocols bgp group *group-name* graceful-restart], or [edit protocols bgp group *group-name* neighbor *neighbor-address* graceful-restart] hierarchy level disables the transmission of the N flag in the graceful restart capability negotiation. The disable-notification-extensions statement at the [edit protocols bgp graceful-restart], [edit protocols bgp group *group-name* graceful-restart], or [edit protocols bgp group *group-name* neighbor *neighbor-address* graceful-restart] hierarchy level also disables the transmission of the N flag in the graceful restart capability negotiation, but in addition, it disables the new rules for invoking graceful restart receiver mode as specified in the

IETF bgp-gr-notification draft, and disables the transmission of the Hard Reset subcode. The Hard Reset subcode is continued to be observed when received in a Notify or a Cease message.

To disable the transmission of N flags and to disable rules for triggering graceful restart at the global or system-wide level:

```
[edit]
protocols {
    bgp {
        graceful-restart {
            disable-notification-flag;
            disable-notification-extensions;
        }
    }
}
```

To disable the transmission of N flags and to disable rules for triggering graceful restart at the group level:

```
[edit]
protocols {
    bgp {
        group group-name {
            graceful-restart {
                disable-notification-flag;
                disable-notification-extensions;
            }
        }
    }
}
```

To disable the transmission of N flags and to disable rules for triggering graceful restart at the neighbor or peer level:

```
[edit]
protocols {
    bgp {
        group group-name {
            graceful-restart {
                disable-notification-flag;
                disable-notification-extensions;
```

```
            }
        }
    }
}
```

SEE ALSO

> Informing the BGP Helper Router or Peer About Retaining Routes By Configuring the Forwarding State Bit for All Address Families and for a Specific Address Family | **1062**

Configuring Long-Lived Graceful Restarter Mode Negotiation for a Specific Address Family in Logical Systems and Routing Instances

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

You can also configure the BGP long-lived graceful restarter mode negotiation mechanism for a particular address family instead of configuring this capability for all address families in a system, logical system, or routing instance. To enable BGP LLGR for a specific address family, include the `graceful-restart long-lived restarter stale-time` *`interval`* statement at one of the following hierarchy levels.

Each routing table is identified by the protocol family or address family indicator (AFI) and a subsequent address family identifier (SAFI). The AFI parameter can be one of the `(l2vpn | inet | route-target)` protocols and the SAFI parameter can be either of the `(flow | labeled-unicast)` protocols for inet family and one of the `(auto-discovery-mspw | auto-discovery-only | signaling)` protcols for L2VPN family..

Configuring LLGR does not require that BGP graceful restart also be configured. The long-lived-graceful-restart section is visible only for families l2vpn, inet labeled-unicast, inet flow and route-target. It is prohibited for inet-mvpn, inet6-mvpn and inet-mdt. It is hidden for other families.

```
[edit logical-systems logical-system-name protocols bgp family inet (labeled-unicast | unicast |
multicast)],
[edit logical-systems logical-system-name protocols bgp group group-name family inet (labeled-
unicast | unicast | multicast)],
[edit logical-systems logical-system-name protocols bgp group group-name neighbor address family
inet (labeled-unicast | unicast | multicast)],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family inet (labeled-unicast | unicast | multicast)],
```

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
group group-name family inet (labeled-unicast | unicast | multicast)],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
group group-name neighbor address family inet (labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp family inet (labeled-unicast |
unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name family inet
(labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name neighbor address
family inet (labeled-unicast | unicast | multicast)]
[edit routing-instances routing-instance-name protocols bgp family inet (labeled-unicast |
unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name family inet
(labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name neighbor address
family inet (labeled-unicast | unicast | multicast)],
```

The stanzas in the per-family graceful-restart long-lived restarter configuration section enables LLGR restarter mode negotiation for BGP globally, or for a group or neighbor. The values are inherited by groups from the global configuration, and by neighbors from the group configuration. The disable attribute is used to override configuration inherited from a higher level. It does not disable LLGR receiver mode; you must disable LLGR receiver mode explicitly for all families as necessary. A hidden `enable` attribute can be used to override an inherited disable attribute. Configuring graceful-restart long-lived restarter at the neighbor level (when it is not configured at the containing group level or globally) causes an internal group to be split. When LLGR restarter is enabled or disabled for a family or the stale- time is changed, the session is reset so that the new capability can be sent to the neighbor.

The range of values for stale-time is from 1 to 16777215 (2^24 – 1) seconds. The value is a simple integer giving the number of seconds by default, but it can also be specified using the following notation:

[<weeks>w][<days>d][<hours>h][<minutes>m][<seconds>s] For example, you can specify 27 days as 27d, 648h, 38880m or 2332800s. 90 minutes can be configured as 1h30m, 90m or 5400s. The specified number of days is multiplied by 86400, the number of hours by 3600 and the number of minutes by 60; these are added to the seconds to get the total. A combined format of days and hours, in different time period units, such as 1d36h are permitted, as long as the specified total does not exceed the maximum stale time.

In addition, times can also be configured using the following notation: <hours>:<minutes>:<seconds> For example, 12:00:00 specifies twelve hours. The hours and minutes are optional.

The two notations can be combined, for example, 2w1d 12:00:02 specifies two weeks, one day, twelve hours and two seconds (1339202 seconds). (Note that the CLI requires double-quotes around a value like this with spaces in it.) Expressed in this notation, the maximum stale time is 27w5d 04:20:15 (27

weeks, 5 days, 4 hours, 20 minutes and 15 seconds). While the show configuration command displays the actually configured values, when the associated timers are displayed in run-time show commands such as `show bgp neighbor`, the values are normalized, such as 1d36h becoming 2d 12:00:00. The full rules for displaying normalized LLGR times depend on the `clear bgp neighbor neighbor-address gracefully` command configuration.

To configure the BGP long-lived graceful restart characteristics per-address family and per-subsequent address family at the global level for a logical system or a routing instance:

Configuring BGP Long-Lived Graceful Restart Per Address Family At the Global Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        graceful-restart {
            long-lived {
                restarter {
                    disable;
                    stale-time interval;
                }
            }
        }
    }
}
```

Configuring BGP Long-Lived Graceful Restart Per Address Family At the Global Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
protocols {
    bgp {
        graceful-restart {
            long-lived {
                restarter {
                    disable;
                    stale-time interval;
                }
            }
        }
```

```
    }
}
```

To configure the BGP long-lived graceful restart characteristics per-address family and per-subsequent address family at the BGP group level for a logical system or a routing instance:

Configuring BGP Long-Lived Graceful Restart Per Address Family At the BGP Group Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        group group-name {
            graceful-restart {
                long-lived {
                    restarter {
                        disable;
                        stale-time interval;
                    }
                }
            }
        }
    }
}
```

Configuring BGP Long-Lived Graceful Restart Per Address Family At the BGP Group Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
protocols {
    bgp {
        group group-name {
            graceful-restart {
                long-lived {
                    restarter {
                        disable;
                        stale-time interval;
                    }
                }
            }
```

```
        }
    }
}
```

To configure the BGP long-lived graceful restart characteristics per-address family and per-subsequent address family at the BGP neighbor group level for a logical system or a routing instance:

Configuring BGP Long-Lived Graceful Restart Per Address Family At the BGP Neighbor Group Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    long-lived {
                        restarter {
                            disable;
                            stale-time interval;
                        }
                    }
                }
            }
        }
    }
}
```

Configuring BGP Long-Lived Graceful Restart Per Address Family At the BGP Neighbor Group Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    long-lived {
                        restarter {
                            disable;
```

```
                        stale-time interval;
                    }
                }
            }
        }
    }
}
```

SEE ALSO

> Increasing the Duration for Preserving BGP Routes Across Slowly-Restarting Peers By BGP Long-Lived Graceful Restart | **1050**

Informing the BGP Helper Router or Peer About Retaining Routes By Configuring the Forwarding State Bit for All Address Families and for a Specific Address Family

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

After a BGP session goes down and before the session is reestablished, stale routes might be retained for up to two consecutive periods, controlled by the restart time and long-lived stale time parameters, respectively. During the first period routing modifications are prevented but with potential null-route filtering of traffic. During the second period, possible null-route filtering of traffic might be reduced but routing changes are visible throughout the network. In your network environment, the setting of the relevant parameters for a particular application must consider the tradeoffs, the network dynamics and potential failure scenarios. If necessary, the first period can be bypassed either by local configuration or by setting the restart time in the graceful restart capability to zero, not listing the address family indicators (AFI) and a subsequent address family identifiers (SAFI) in that capability.

The setting of the F bit (and the "Forwarding State" bit of the accompanying GR capability) depends in part on deployment considerations. The F bit can be interpreted to indicate the helper router needs to flush associated routes (if the bit is left clear). An important scenario in which LLGR is used is for routes that are more similar to configuration than to traditional routing (hop-by-hop forwarding instead of tunnel-based routing). For such routes, it might be useful to always set the F bit, regardless of other considerations. Similarly, for control-plane-only entities such as dedicated route reflectors, that do not participate in the forwarding plane, it is preferred that the F bit be always set. Overall, the guideline to

be adopted is that if loss of state on the restarting router can reasonably be expected to cause a forwarding loop or null route, the F bit must be set judiciously, depending on whether state has been retained. You can determine whether the F bit needs to be set or not, based on your deployment needs and configured settings. It might be necessary to advertise stale routes to a CE in some VPN deployments, even if the CE does not support this specification. In such a scenario, the network operator configuring their PE to advertise such routes must notify the operator of the CE receiving the routes, and the CE must be configured to depreference the routes. Typically, BGP implementations perform this behavior by matching on the LLGR_STALE community, and setting the LOCAL_PREF for matching routes to zero.

You can specify the Forwarding State bit, which is a BGP configuration option that can be defined at the global, group and neighbor levels, for any logical system or routing instance. To specify the Forwarding State bit at the global, BGP group, or BGP neighbor level, include the `forwarding-state-bit (as-rr-client | from-fib)` statement at the `[edit protocols bgp graceful-restart]`, `[edit protocols bgp group-group-name graceful-restart]`, or `[edit protocols bgp group-group-name neighbor neighbor-address graceful-restart]` hierarchy level. The forwarding-state-bit attribute controls how the Forwarding State bit is set in both graceful restart and long-lived graceful restart capability advertisements. By default, the value depends on whether the neighbor is a route reflector client. If the neighbor is not a route reflector client, the value is set according to the state of the associated FIB in compliance with RFC 4724. If the neighbor is a route reflector client, the value is set to 1 for all families except inet unicast and inet6 unicast, which use the state of the associated FIB. The `as-rr-client` option sets the behavior for all address families to be the same as the functionality for a route reflector client. The `from-fib` option forces the behavior for all address families to be as they would be for a non-route-reflector client.

To configure the forwarding-state flag negotiation at the global level:

```
[edit]
protocols {
    bgp {
        graceful-restart {
            forwarding-state-bit (as-rr-client | from-fib);
        }
    }
}
```

To configure the forwarding-state flag negotiation at the group level:

```
[edit]
protocols {
    bgp {
        group group-name {
            graceful-restart {
```

```
                forwarding-state-bit (as-rr-client | from-fib);
            }
        }
    }
}
```

To configure the forwarding-state flag negotiation at the neighbor or peer group level:

```
[edit]
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    forwarding-state-bit (as-rr-client | from-fib);
                }
            }
        }
    }
}
```

In addition to the global setting for the Forwarding State bit, the Forwarding State bit behavior can be specified for individual families. Changing the forwarding-state-bit setting has no effect on any existing sessions. To specify the Forwarding State bit for a particular address family, include the `forwarding-state-bit (set | from-fib)` statement at the `[edit protocols bgp graceful-restart family address-family subsequent-address-family]`, `[edit protocols bgp group-group-name graceful-restart family address-family subsequent-address-family]`, or `[edit protocols bgp group-group-name neighbor neighbor-address graceful-restart family address-family subsequent-address-family]` hierarchy level on a logical system and a routing instance. Per-family BGP configuration options are added to control the Forwarding State bit in graceful restart and long-lived graceful restart capability advertisements. They can be specified for the default logical system or for a specific logical system, and for the primary routing instance or a specific routing instance. The `per-family forwarding-state-bit` attribute overrides the default rules or the global configuration for setting the Forwarding State bit. The `set` option forces the Forwarding State bit to be set to 1. The `from-fib` option causes the value to be set according to the state of the associated FIB. Changing the per-family forwarding-state-bit setting has no effect on any existing sessions.

The following are the complete configuration hierarchy levels at which you can include the `forwarding-state-bit (set | from-fib)` statement to configure the forwarding state bit per address family:

```
[edit logical-systems logical-system-name protocols bgp family inet (labeled-unicast | unicast |
multicast)],
```

```
[edit logical-systems logical-system-name protocols bgp group group-name family inet (labeled-
unicast | unicast | multicast)],
[edit logical-systems logical-system-name protocols bgp group group-name neighbor address family
inet (labeled-unicast | unicast | multicast)],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family inet (labeled-unicast | unicast | multicast)],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
group group-name family inet (labeled-unicast | unicast | multicast)],
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
group group-name neighbor address family inet (labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp family inet (labeled-unicast |
unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name family inet
(labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name neighbor address
family inet (labeled-unicast | unicast | multicast)]
[edit routing-instances routing-instance-name protocols bgp family inet (labeled-unicast |
unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name family inet
(labeled-unicast | unicast | multicast)],
[edit routing-instances routing-instance-name protocols bgp group group-name neighbor address
family inet (labeled-unicast | unicast | multicast)],
```

To configure the forwarding state bit for BGP long-lived graceful restart per-address family and per-subsequent address family at the global level for a logical system or a routing instance:

Configuring the Forwarding State Bit Per Address Family At the Global Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        graceful-restart {
            forwarding-state-bit (set | from-fib);
        }
    }
}
```

Configuring the Forwarding State Bit Per Address Family At the Global Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
```

```
protocols {
    bgp {
        graceful-restart {
            forwarding-state-bit (set | from-fib);
        }
    }
}
```

To configure the forwarding state bit for BGP long-lived graceful restart per-address family and per-subsequent address family at the BGP group level for a logical system or a routing instance:

Configuring the Forwarding State Bit Per Address Family At the BGP Group Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        group group-name {
            graceful-restart {
                forwarding-state-bit (set | from-fib);
            }
        }
    }
}
```

Configuring the Forwarding State Bit Per Address Family At the BGP Group Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
protocols {
    bgp {
        group group-name {
            graceful-restart {
                forwarding-state-bit (set | from-fib);
            }
        }
    }
}
```

To configure the forwarding state bit for BGP long-lived graceful restart per-address family and per-subsequent address family at the BGP neighbor group level for a logical system or a routing instance:

Configuring the Forwarding State Bit Per Address Family At the BGP Neighbor Group Level for Logical Systems

```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols bgp
family address-family subsequent-address-family
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    forwarding-state-bit (set | from-fib);
                }
            }
        }
    }
}
```

Configuring the Forwarding State Bit Per Address Family At the BGP Neighbor Group Level for Routing Instances

```
[edit routing-instances routing-instance-name protocols bgp family address-family subsequent-
address-family
protocols {
    bgp {
        group group-name {
            neighbor neighbor-address {
                graceful-restart {
                    forwarding-state-bit (set | from-fib);
                }
            }
        }
    }
}
```

SEE ALSO

Understanding Maximum Period Configuration for Automatic Generation of BGP Keepalives by Kernel Timers After Switchover | **1044**

Example: Preserving Route Details for Slow and Latent BGP Peers By Using BGP Long-Lived Graceful Restart

IN THIS SECTION

- Requirements | 1068
- Overview | 1069
- Configuration | 1070
- Verification | 1073

Junos OS supports the mechanism to preserve BGP routing details for a longer period from a failed BGP peer than the duration for which such routing information is maintained using the BGP graceful restart functionality.

Historically, routing protocols and BGP, in particular, have been designed with a focus on correctness, where a significant aspect of the "correctness" is for each network element's forwarding state to converge toward the current state of the network as quickly as possible. For this reason, the protocol was designed to remove state advertised by routers which went down (from a BGP perspective) as promptly as possible. Using BGP Graceful Restart defined in RFC 4724, the fast convergence functionality has been an attempt to rapidly remove "stale" state from the network.

BGP long-lived graceful restart (LLGR) allows a network operator to choose to maintain stale routing information from a failed BGP peer much longer than the existing BGP Graceful Restart facility. This functionality to maintain the BGP routes for a longer time period is in accordance with the IETF draft, *Support for Long-lived BGP Graceful Restart—draft-uttaro-idr-bgp-persistence-03*. According to this draft, long-lived graceful restart (LLGR) must be explicitly configured per NLRI, and it includes provisions to prevent the spread of stale information to other peers that do not recognize and validate LLGR.

This example describes how to configure BGP long-lived graceful restart functionality on MX Series routers, and contains the following sections:

Requirements

This example uses the following hardware and software components:

- One MX Series router with an MPC.
- Junos OS Release 15.1R1 or later for MX Series routers

Before you configure BGP long-lived graceful restart, make sure you:

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

Overview

IN THIS SECTION

- Topology | **1069**

Graceful restart allows a routing device undergoing a restart to inform its adjacent neighbors and peers of its condition. During a graceful restart, the restarting device and its neighbors continue forwarding packets without disrupting network performance. Because neighboring devices assist in the restart (these neighbors are called *helper routers*), the restarting device can quickly resume full operation without recalculating algorithms.

Long-lived graceful restart receiver mode is enabled by default, unless ordinary graceful restart receiver mode is disabled. To enable the BGP long-lived graceful restart (LLGR) capability, include the `long-lived receiver enable` statement at the `[edit protocols bgp graceful-restart]` hierarchy level. Apart from enabling BGP LLGR at the global or system-wide level, you can also include the long-lived receiver enable statement at the `[edit protocols bgp group` *`group-name`* `graceful-restart]` hierarchy level to configure LLGR for a particular BGP group and at the `[edit protocols bgp group` *`group-name`* `neighbor` *`neighbor-address`* `graceful-restart]` hierarchy level to configure LLGR for a particular BGP neighbor. To disable the BGP LLGR mechanism, include the `long-lived receiver disable` option the `[edit protocols bgp graceful-restart]`, `[edit protocols bgp group` *`group-name`* `graceful-restart]`, or [edit protocols bgp group-group-name neighbor neighbor-address graceful-restart] hierarchy level. Disabling LLGR deactivates all of the LLGR capabilities (both receiver and restarter modes) for all NLRI families. This property is inherited by groups from the global configuration, and by neighbors from the group configuration.

Topology

Consider a sample scenario in which you want to increase the time period for which stale routes are maintained for a BGP peer or neighbor with the address of 1.2.3.4. Besides specifying the duration for which the routes must be retained for stale sessions and when a graceful restart of a peer occurs, you can also configure BGP routers from certain address prefixes to be disregarded when you define the long-lived graceful restart mechanism. You can define a list of IPv4 or IPv6 address prefixes for use in a routing policy statement and a BGP community to be included in the routing policy. If you set the action modifier to reject routes from a particular prefix, such BGP routes are not maintained for the increased time period.

You can also configure the BGP long-lived graceful restarter mode negotiation mechanism for a particular address family instead of configuring this capability for all address families in a system, logical system, or routing instance. To enable BGP LLGR for a specific address family, include the `graceful-restart long-lived restarter stale-time` *`interval`* statement at one of the following hierarchy levels.

Each routing table is identified by the protocol family or address family indicator (AFI) and a subsequent address family identifier (SAFI). The AFI parameter can be one of the `(l2vpn | inet | route-target)` protocols and the SAFI parameter can be either of the `(flow | labeled-unicast)` protocols for inet family and one of the `(auto-discovery-mspw | auto-discovery-only | signaling)` protcols for L2VPN family..

Configuring LLGR does not require that BGP graceful restart also be configured. The long-lived-graceful-restart section is visible only for families l2vpn, inet labeled-unicast, inet flow and route-target. It is prohibited for inet-mvpn, inet6-mvpn and inet-mdt. It is hidden for other families.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1070**
- Configuring Long-Lived Graceful Restart for Restarter Mode | **1071**
- Results | **1072**

CLI Quick Configuration

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

Configuring the Address Prefix List, BGP Community, and BGP Routing Policy

```
set policy-options prefix-list special 44.44.44.44/32
set policy-options community llgr-community llgr-stale
set policy-options policy-statement llgr-import from prefix-list special
set policy-options policy-statement llgr-import from community llgr-community
set policy-options policy-statement llgr-import then reject
```

Configuring the BGP Group, NLRI, and Long-Lived Graceful Restart

```
set protocols bgp group ibgp-group type internal
set protocols bgp group ibgp-group import llgr-import
set protocols bgp group ibgp-group family inet unicast
set protocols bgp group ibgp-group family inet unicast graceful-restart long-lived restarter
stale-time 12h
```

Configuring the BGP Neighbor Group

```
set protocols bgp group ibgp-group neighbor 1.2.3.4
```

Configuring Long-Lived Graceful Restart for Restarter Mode

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.

1. Configure the address prefix list, BGP community, and the match condition and action modifier for the BGP routing policy.

```
[edit]
user@ host# set policy-options prefix-list special 44.44.44.44/32
user@ host# set policy-options community llgr-community llgr-stale
user@ host# set policy-options policy-statement llgr-import from prefix-list special
user@ host# set policy-options policy-statement llgr-import from community llgr-community
user@ host# set policy-options policy-statement llgr-import then reject
```

2. Configure the BGP group, address family, and long-lived graceful restart functionality for restarter mode with the stale time for flows.

```
[edit]
user@ host# set protocols bgp group ibgp-group type internal
user@ host# set protocols bgp group ibgp-group import llgr-import
user@ host# set protocols bgp group ibgp-group family inet unicast
```

```
user@ host# set protocols bgp group ibgp-group family inet unicast graceful-restart long-
lived restarter stale-time 12h
```

3. Configure the BGP neighbor group.

```
[edit]
user@ host# set protocols bgp group ibgp-group neighbor 1.2.3.4
```

Results

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

```
[edit]
```

```
user@host# show policy-options
policy-options {
    prefix-list special 44.44.44.44/32;
    community llgr-community llgr-stale;
    policy-statement llgr-import {
        from {
            prefix-list special;
            community llgr-community;
        }
        then {
            reject;
        }
    }
}
```

```
user@host# show protocols
protocols {
    bgp {
        group ibgp-group {
            type internal;
            import llgr-import;
            family inet unicast {
```

```
                graceful-restart {
                    long-lived {
                        restarter {
                            stale-time 12h;
                        }
                    }
                }
            }
            neighbor 1.2.3.4;
        }
    }
}
```

Verification

IN THIS SECTION

- Verifying That the Long-Lived Graceful Restart Capability is Enabled | **1073**

Confirm that the configuration is working properly.

Verifying That the Long-Lived Graceful Restart Capability is Enabled

Purpose

Verify the BGP long-lived graceful restart capability configured for BGP neighbor level

Action

While LLGR receiver mode is active (a peer that negotiated LLGR has disconnected and not yet reconnected), the output of the `show bgp neighbor` command displays the amount of time left until the LLGR expires, the time remaining on the GR stale timer, and RIB details:

```
user@router> show bgp neighbor
Peer: 10.4.12.11 AS 100        Local: 10.6.128.225 AS 100
  Type: Internal    State: Active         Flags: <>
  Last State: Idle          Last Event: Start
  Last Error: None
```

```
    Export: [ foo ]
    Options: <Preference LocalAddress Refresh GracefulRestart>
    Options: <LLGR>
    Local Address: 10.6.128.225 Holdtime: 90 Preference: 170
    Number of flaps: 3
    Last flap event: Restart
    Error: 'Cease' Sent: 0 Recv: 1
    Time until long-lived stale routes deleted: inet-vpn-unicast 10:00:22 route-target 10:00:22
    Table bgp.l3vpn.0
      RIB State: BGP restart is complete
      RIB State: VPN restart is complete
      Send state: not advertising
      Active prefixes:              0
      Received prefixes:            7
      Accepted prefixes:            7
      Suppressed due to damping:    0
    Table foo.inet.0 Bit: 30000
      RIB State: BGP restart is complete
      RIB State: VPN restart is complete
      Send state: not in sync
      Active prefixes:              0
      Received prefixes:            7
      Accepted prefixes:            7
      Suppressed due to damping:    0
```

Meaning

The output shows information about BGP neighbors.

CHAPTER 7

Configuring Multiprotocol for a BGP Session

IN THIS CHAPTER

- Multiprotocol BGP | **1076**

Multiprotocol BGP

IN THIS SECTION

- Understanding Multiprotocol BGP | **1076**
- Example: Configuring IPv6 BGP Routes over IPv4 Transport | **1084**
- Advertising IPv4 Routes over BGP IPv6 Sessions Overview | **1094**
- Example: Advertising IPv4 Routes over IPv6 BGP Sessions | **1095**
- Understanding Redistribution of IPv4 Routes with IPv6 Next Hop into BGP | **1104**
- Configuring BGP to Redistribute IPv4 Routes with IPv6 Next-Hop Addresses | **1111**
- Enabling Layer 2 VPN and VPLS Signaling | **1114**
- Understanding BGP Flow Routes for Traffic Filtering | **1115**
- Example: Enabling BGP to Carry Flow-Specification Routes | **1123**
- Example: Configuring BGP to Carry IPv6 Flow Specification Routes | **1145**
- Configuring BGP Flow Specification Action Redirect to IP to Filter DDoS Traffic | **1158**
- Forwarding Traffic Using BGP Flow Specification DSCP Action | **1162**
- Configuring Policies for Flow Route Validation | **1163**

Understanding Multiprotocol BGP

IN THIS SECTION

- Limiting the Number of Prefixes Received on a BGP Peer Session | **1081**
- Limiting the Number of Prefixes Accepted on a BGP Peer Session | **1082**
- Configuring BGP Routing Table Groups | **1083**
- Resolving Routes to PE Routing Devices Located in Other ASs | **1084**
- Allowing Labeled and Unlabeled Routes | **1084**

Multiprotocol BGP (MP-BGP) is an extension to BGP that enables BGP to carry routing information for multiple network layers and address families. MP-BGP can carry the unicast routes used for multicast routing separately from the routes used for unicast IP forwarding.

To enable MP-BGP, you configure BGP to carry network layer reachability information (NLRI) for address families other than unicast IPv4 by including the `family inet` statement:

```
family inet {
    (any | flow | labeled-unicast | multicast | unicast) {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
}
        <loops number>;
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
 }
        rib-group group-name;
        topology name {
            community {
                target identifier;
            }
        }
    }
}
```

To enable MP-BGP to carry NLRI for the IPv6 address family, include the `family inet6` statement:

```
family inet6 {
    (any | labeled-unicast | multicast | unicast) {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
}
        <loops number>;
```

```
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
 }
        rib-group group-name;
    }
}
```

On routers only, to enable MP-BGP to carry Layer 3 virtual private network (VPN) NLRI for the IPv4 address family, include the `family inet-vpn` statement:

```
family inet-vpn {
    (any | flow | multicast | unicast) {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
}
        <loops number>;
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}
}
        rib-group group-name;
    }
}
```

On routers only, to enable MP-BGP to carry Layer 3 VPN NLRI for the IPv6 address family, include the `family inet6-vpn` statement:

```
family inet6-vpn {
    (any | multicast | unicast) {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
```

```
            hide-excess <percentage>;}
}
        <loops number>;
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}}
        rib-group group-name;
    }
}
```

On routers only, to enable MP-BGP to carry multicast VPN NLRI for the IPv4 address family and to enable VPN signaling, include the `family inet-mvpn` statement:

```
family inet-mvpn {
    signaling {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}}
        <loops number>;
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}}
    }
}
```

To enable MP-BGP to carry multicast VPN NLRI for the IPv6 address family and to enable VPN signaling, include the `family inet6-mvpn` statement:

```
family inet6-mvpn {
    signaling {
        accepted-prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout (forever | minutes)>;
            drop-excess <percentage>;
            hide-excess <percentage>;}}
```

```
        <loops number>;
        prefix-limit {
            maximum number;
            teardown <percentage> <idle-timeout <forever | minutes>;
            drop-excess <percentage>;
            hide-excess <percentage>;}}
    }
}
```

For more information about multiprotocol BGP-based multicast VPNs, see the Junos OS Multicast Protocols User Guide.

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

NOTE: If you change the address family specified in the `[edit protocols bgp family]` hierarchy level, all current BGP sessions on the routing device are dropped and then reestablished.

In Junos OS Release 9.6 and later, you can specify a loops value for a specific BGP address family.

By default, BGP peers carry only unicast routes used for unicast forwarding purposes. To configure BGP peers to carry only multicast routes, specify the `multicast` option. To configure BGP peers to carry both unicast and multicast routes, specify the `any` option.

When MP-BGP is configured, BGP installs the MP-BGP routes into different routing tables. Each routing table is identified by the protocol family or address family indicator (AFI) and a subsequent address family identifier (SAFI).

The following list shows all possible AFI and SAFI combinations:

- AFI=1, SAFI=1, IPv4 unicast
- AFI=1, SAFI=2, IPv4 multicast
- AFI=1, SAFI=128, L3VPN IPv4 unicast
- AFI=1, SAFI=129, L3VPN IPv4 multicast
- AFI=2, SAFI=1, IPv6 unicast
- AFI=2, SAFI=2, IPv6 multicast
- AFI=25, SAFI=65, BGP-VPLS/BGP-L2VPN
- AFI=2, SAFI=128, L3VPN IPv6 unicast

- AFI=2, SAFI=129, L3VPN IPv6 multicast
- AFI=1, SAFI=132, RT-Constrain
- AFI=1, SAFI=133, Flow-spec
- AFI=1, SAFI=134, Flow-spec
- AFI=3, SAFI=128, CLNS VPN
- AFI=1, SAFI=5, NG-MVPN IPv4
- AFI=2, SAFI=5, NG-MVPN IPv6
- AFI=1, SAFI=66, MDT-SAFI
- AFI=1, SAFI=4, labeled IPv4
- AFI=2, SAFI=4, labeled IPv6 (6PE)

Routes installed in the inet.2 routing table can only be exported to MP-BGP peers because they use the SAFI, identifying them as routes to multicast sources. Routes installed in the inet.0 routing table can only be exported to standard BGP peers.

The inet.2 routing table should be a subset of the routes that you have in inet.0, since it is unlikely that you would have a route to a multicast source to which you could not send unicast traffic. The inet.2 routing table stores the unicast routes that are used for multicast reverse-path-forwarding checks and the additional reachability information learned by MP-BGP from the NLRI multicast updates. An inet.2 routing table is automatically created when you configure MP-BGP (by setting NLRI to `any`).

When you enable MP-BGP, you can do the following:

Limiting the Number of Prefixes Received on a BGP Peer Session

You can limit the number of prefixes received on a BGP peer session, and log rate-limited messages when the number of injected prefixes exceeds a set limit. You can also tear down the peering when the number of prefixes exceeds the limit.

To configure a limit to the number of prefixes that can be received on a BGP session, include the `prefix-limit` statement:

```
prefix-limit {
    maximum number;
    teardown <percentage> <idle-timeout (forever | minutes)>;
    drop-excess <percentage>;
```

```
    hide-excess <percentage>;
}
```

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

For `maximum` *number*, specify a value in the range from 1 through 4,294,967,295. When the specified maximum number of prefixes is exceeded, a system log message is sent.

If you include the `teardown` statement, the session is torn down when the maximum number of prefixes is exceeded. If you specify a percentage, messages are logged when the number of prefixes exceeds that percentage of the specified maximum limit. After the session is torn down, it is reestablished in a short time (unless you include the `idle-timeout` statement). If you include the `idle-timeout` statement, the session can be kept down for a specified amount of time, or forever. If you specify `forever`, the session is reestablished only after the you issue a `clear bgp neighbor` command. If you include the `drop-excess` <*percentage*> option, the excess routes are dropped when the maximum number of prefixes is reached. If you specify a percentage, the routes are logged when the number of prefixes exceeds that percentage value of the maximum number. If you include the `hide-excess` <*percentage*> option, the excess routes are hidden when the maximum number of prefixes is reached. If you specify a percentage, the routes are logged when the number of prefixes exceeds that percentage value of the maximum number. If the percentage is modified, the routes are re-evaluated automatically. If the active routes drop below the specified percentage, those routes are kept as hidden.

NOTE: In Junos OS Release 9.2 and later, you can alternatively configure a limit to the number of prefixes that can be accepted on a BGP peer session. For more information, see "Limiting the Number of Prefixes Accepted on a BGP Peer Session" on page 1082.

Limiting the Number of Prefixes Accepted on a BGP Peer Session

In Junos OS Release 9.2 and later, you can limit the number of prefixes that can be accepted on a BGP peer session. When that specified limit is exceeded, a system log message is sent. You can also specify to reset the BGP session if the limit to the number of specified prefixes is exceeded.

To configure a limit to the number of prefixes that can be accepted on a BGP peer session, include the `accepted-prefix-limit` statement:

```
accepted-prefix-limit {
    maximum number;
    teardown <percentage> <idle-timeout (forever | minutes)>;
    drop <percentage>;
```

```
    hide <percentage>;
}
```

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

For **maximum *number***, specify a value in the range from 1 through 4,294,967,295.

Include the `teardown` statement to reset the BGP peer session when the number of accepted prefixes exceeds the configured limit. You can also include a percentage value from 1 through 100 to have a system log message sent when the number of accepted prefixes exceeds that percentage of the maximum limit. By default, a BGP session that is reset is reestablished within a short time. Include the `idle-timeout` statement to prevent the BGP session from being reestablished for a specified period of time. You can configure a timeout value from 1 through 2400 minutes. Include the **forever** option to prevent the BGP session from being reestablished until you issue the `clear bgp neighbor` command. If you include the `drop-excess <percentage>` statement and specify a percentage, the excess routes are dropped when the number of prefixes exceeds the percentage. If you include the `hide-excess <percentage>` statement and specify a percentage, the excess routes are hidden when the number of prefixes exceeds the percentage. If the percentage is modified, the routes are re-evaluated automatically.

NOTE: When nonstop active routing (NSR) is enabled and a switchover to a backup Routing Engine occurs, BGP peers that are down are automatically restarted. The peers are restarted even if the `idle-timeout forever` statement is configured.

NOTE: Alternatively, you can configure a limit to the number of prefixes that can be *received* (as opposed to accepted) on a BGP peer session. For more information, see "Limiting the Number of Prefixes Received on a BGP Peer Session" on page 1081.

Configuring BGP Routing Table Groups

When a BGP session receives a unicast or multicast NLRI, it installs the route in the appropriate table (**inet.0** or **inet6.0** for unicast, and **inet.2** or **inet6.2** for multicast). To add unicast prefixes to both the unicast and multicast tables, you can configure BGP routing table groups. This is useful if you cannot perform multicast NLRI negotiation.

To configure BGP routing table groups, include the `rib-group` statement:

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

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

Resolving Routes to PE Routing Devices Located in Other ASs

You can allow labeled routes to be placed in the **inet.3** routing table for route resolution. These routes are then resolved for provider edge (PE) routing device connections where the remote PE is located across another autonomous system (AS). For a PE routing device to install a route in the VPN routing and forwarding (VRF) routing instance, the next hop must resolve to a route stored within the **inet.3** table.

To resolve routes into the **inet.3** routing table, include the `resolve-vpn` statement:

```
resolve-vpn group-name;
```

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

Allowing Labeled and Unlabeled Routes

You can allow both labeled and unlabeled routes to be exchanged in a single session. The labeled routes are placed in the inet.3 or inet6.3 routing table, and both labeled and unlabeled unicast routes can be sent to or received by the routing device.

To allow both labeled and unlabeled routes to be exchanged, include the `rib` statement:

```
rib  (inet.3 | inet6.3);
```

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

Example: Configuring IPv6 BGP Routes over IPv4 Transport

IN THIS SECTION

- Requirements | **1085**
- Overview | **1085**
- Configuration | **1086**
- Verification | **1091**

This example demonstrates how to export both IPv6 and IPv4 prefixes over an IPv4 connection where both sides are configured with an IPv4 interface.

Requirements

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

Overview

Keep the following in mind when exporting IPv6 BGP prefixes:

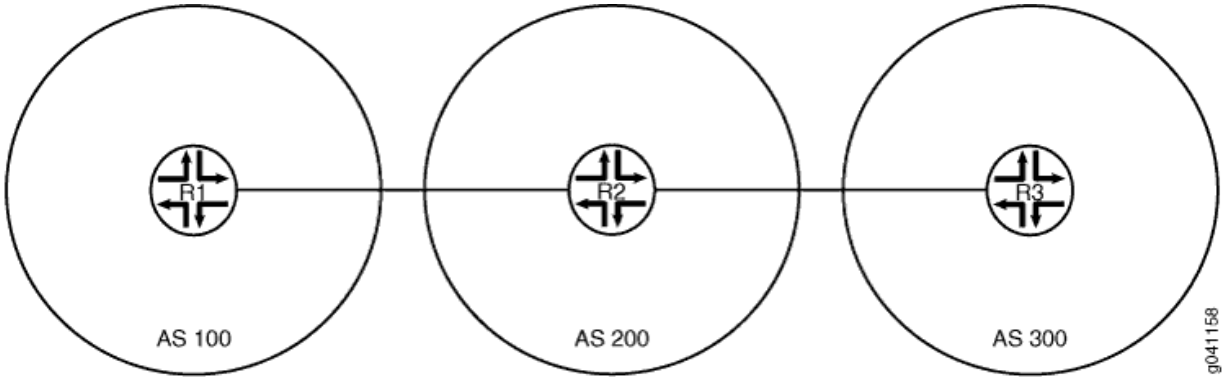
- BGP derives next-hop prefixes using the IPv4-mapped IPv6 prefix. For example, the IPv4 next-hop prefix `10.19.1.1` translates to the IPv6 next-hop prefix ::ffff:10.19.1.1.

  NOTE: There must be an active route to the IPv4-mapped IPv6 next hop to export IPv6 BGP prefixes.

- An IPv6 connection must be configured over the link. The connection must be either an IPv6 tunnel or a dual-stack configuration. Dual stacking is used in this example.
- When configuring IPv4-mapped IPv6 prefixes, use a mask that is longer than 96 bits.
- Configure a static route if you want to use normal IPv6 prefixes. This example uses static routes.

Figure 69 on page 1085 shows the sample topology.

Figure 69: Topology for Configuring IPv6 BGP Routes over IPv4 Transport



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1086
- Configuring Device R1 | 1088

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 R1

```
set interfaces fe-1/2/0 unit 1 family inet address 192.168.10.1/24
set interfaces fe-1/2/0 unit 1 family inet6 address ::ffff:192.168.10.1/120
set interfaces lo0 unit 1 family inet address 10.10.10.1/32
set protocols bgp group ext type external
set protocols bgp group ext family inet unicast
set protocols bgp group ext family inet6 unicast
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 192.168.10.10
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options rib inet6.0 static route ::ffff:192.168.20.0/120 next-
hop ::ffff:192.168.10.10
set routing-options static route 192.168.20.0/24 next-hop 192.168.10.10
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 192.168.10.10/24
set interfaces fe-1/2/0 unit 2 family inet6 address ::ffff:192.168.10.10/120
```

```
set interfaces fe-1/2/1 unit 3 family inet address 192.168.20.21/24
set interfaces fe-1/2/1 unit 3 family inet6 address ::ffff:192.168.20.21/120
set interfaces lo0 unit 2 family inet address 10.10.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext family inet unicast
set protocols bgp group ext family inet6 unicast
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext neighbor 192.168.10.1 peer-as 100
set protocols bgp group ext neighbor 192.168.20.1 peer-as 300
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/0 unit 4 family inet address 192.168.20.1/24
set interfaces fe-1/2/0 unit 4 family inet6 address ::ffff:192.168.20.1/120
set interfaces lo0 unit 3 family inet address 10.10.20.1/32
set protocols bgp group ext type external
set protocols bgp group ext family inet unicast
set protocols bgp group ext family inet6 unicast
set protocols bgp group ext export send-direct
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 192.168.20.21
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options rib inet6.0 static route ::ffff:192.168.10.0/120 next-
hop ::ffff:192.168.20.21
set routing-options static route 192.168.10.0/24 next-hop 192.168.20.21
set routing-options autonomous-system 300
```

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces, including both an IPv4 address and an IPv6 address.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 192.168.10.1/24
user@R1# set fe-1/2/0 unit 1 family inet6 address ::ffff:192.168.10.1/120
user@R1# set lo0 unit 1 family inet address 10.10.10.1/32
```

2. Configure EBGP.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set export send-direct
user@R1# set export send-static
user@R1# set peer-as 200
user@R1# set neighbor 192.168.10.10
```

3. Enable BGP to carry IPv4 unicast and IPv6 unicast routes.

```
[edit protocols bgp group ext]
user@R1# set family inet unicast
user@R1# set family inet6 unicast
```

IPv4 unicast routes are enabled by default. However, when you configure other NLRI address families, IPv4 unicast must be explicitly configured.

4. Configure the routing policy.

```
[edit policy-options]
user@R1# set policy-statement send-direct term 1 from protocol direct
user@R1# set policy-statement send-direct term 1 then accept
```

```
user@R1# set policy-statement send-static term 1 from protocol static
user@R1# set policy-statement send-static term 1 then accept
```

5. Configure some static routes.

```
[edit routing-options]
user@R1# set rib inet6.0 static route ::ffff:192.168.20.0/120 next-hop ::ffff:192.168.10.10
user@R1# set static route 192.168.20.0/24 next-hop 192.168.10.10
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 100
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 192.168.10.1/24;
        }
        family inet6 {
            address ::ffff:192.168.10.1/120;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.10.10.1/32;
        }
```

```
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
```

```
user@R1# show protocols
bgp {
    group ext {
        type external;
        family inet {
            unicast;
        }
        family inet6 {
            unicast;
        }
        export [ send-direct send-static ];
        peer-as 200;
        neighbor 192.168.10.10;
    }
}
```

```
user@R1# show routing-options
rib inet6.0 {
    static {
        route ::ffff:192.168.20.0/120 next-hop ::ffff:192.168.10.10;
    }
}
```

```
static {
    route 192.168.20.0/24 next-hop 192.168.10.10;
}
autonomous-system 100;
```

If you are done configuring the device, enter **commit** from configuration mode. Repeat the configuration on Device R2 and Device R3, changing the interface names and IP addresses, as needed.

Verification

IN THIS SECTION

- Checking the Neighbor Status | **1091**
- Checking the Routing Table | **1094**

Confirm that the configuration is working properly.

Checking the Neighbor Status

Purpose

Make sure that BGP is enabled to carry IPv6 unicast routes.

Action

From operational mode, enter the `show bgp neighbor` command.

```
user@R2> show bgp neighbor
Peer: 192.168.10.1+179 AS 100  Local: 192.168.10.10+54226 AS 200
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct send-static ]
  Options: <Preference AddressFamily PeerAS Refresh>
  Address families configured: inet-unicast inet6-unicast
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 10.10.10.1      Local ID: 10.10.0.1         Active Holdtime: 90
```

```
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/2/0.2
  NLRI for restart configured on peer: inet-unicast inet6-unicast
  NLRI advertised by peer: inet-unicast inet6-unicast
  NLRI for this session: inet-unicast inet6-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-unicast inet6-unicast
  NLRI of received end-of-rib markers: inet-unicast inet6-unicast
  NLRI of all end-of-rib markers sent: inet-unicast inet6-unicast
  Peer supports 4 byte AS extension (peer-as 100)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            3
    Accepted prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          4
  Table inet6.0 Bit: 20000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 24   Sent 12   Checked 60
  Input messages:  Total 132    Updates 6       Refreshes 0     Octets 2700
  Output messages: Total 133    Updates 3       Refreshes 0     Octets 2772
  Output Queue[0]: 0
  Output Queue[1]: 0

Peer: 192.168.20.1+179 AS 300  Local: 192.168.20.21+54706 AS 200
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-direct send-static ]
  Options: <Preference AddressFamily PeerAS Refresh>
  Address families configured: inet-unicast inet6-unicast
```

```
Holdtime: 90 Preference: 170
Number of flaps: 0
Peer ID: 10.10.20.1      Local ID: 10.10.0.1         Active Holdtime: 90
Keepalive Interval: 30         Peer index: 1
BFD: disabled, down
Local Interface: fe-1/2/1.3
NLRI for restart configured on peer: inet-unicast inet6-unicast
NLRI advertised by peer: inet-unicast inet6-unicast
NLRI for this session: inet-unicast inet6-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-unicast inet6-unicast
NLRI of received end-of-rib markers: inet-unicast inet6-unicast
NLRI of all end-of-rib markers sent: inet-unicast inet6-unicast
Peer supports 4 byte AS extension (peer-as 300)
Peer does not support Addpath
Table inet.0 Bit: 10000
  RIB State: BGP restart is complete
  Send state: in sync
  Active prefixes:              1
  Received prefixes:            3
  Accepted prefixes:            2
  Suppressed due to damping:    0
  Advertised prefixes:          4
Table inet6.0 Bit: 20000
  RIB State: BGP restart is complete
  Send state: in sync
  Active prefixes:              0
  Received prefixes:            1
  Accepted prefixes:            1
  Suppressed due to damping:    0
  Advertised prefixes:          2
Last traffic (seconds): Received 1    Sent 15   Checked 75
Input messages:  Total 133    Updates 6       Refreshes 0     Octets 2719
Output messages: Total 131    Updates 3       Refreshes 0     Octets 2734
Output Queue[0]: 0
Output Queue[1]: 0
```

Meaning

The various occurrences of **inet6-unicast** in the output shows that BGP is enabled to carry IPv6 unicast routes.

Checking the Routing Table

Purpose

Make sure that Device R2 has BGP routes in its inet6.0 routing table.

Action

From operational mode, enter the `show route protocol bgp inet6.0` command.

```
user@R2> show route protocol bgp table inet6.0
inet6.0: 7 destinations, 10 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

::ffff:192.168.10.0/120  [BGP/170] 01:03:49, localpref 100, from 192.168.20.1
                      AS path: 300 I
                    > to ::ffff:192.168.20.21 via fe-1/2/1.3
::ffff:192.168.20.0/120  [BGP/170] 01:03:53, localpref 100, from 192.168.10.1
                      AS path: 100 I
                    > to ::ffff:192.168.10.10 via fe-1/2/0.2
```

SEE ALSO

Understanding Multiprotocol BGP | **1076**

Advertising IPv4 Routes over BGP IPv6 Sessions Overview

In an IPv6 network, BGP typically advertises IPv6 network layer reachability information over an IPv6 session between BGP peers. In earlier releases, Junos OS supported the exchange of inet6 unicast, inet6 multicast, or inet6 labeled-unicast address families only. This feature allows the exchange of all BGP address families. In a dual-stack environment that has IPv6 in its core. this feature enables BGP to advertise IPv4 unicast reachability with IPv4 next hop over an IPv6 BGP session.

This feature is for BGP IPv6 sessions only, where IPv4 is configured at both endpoints. The `local-ipv4-address` can be a loopback address or any ipv4 address for an IBGP or multiple-hop EBGP session. For single-hop external BGP speakers that are not part of BGP confederations, if the configured local IPv4 address is not directly connected, the BGP session is closed and remains idle and an error is generated, which is displayed in the output of the `show bgp neighbor` command.

To enable IPv4 route advertising over IPv6 session, configure `local-ipv4-address` as follows:

```
[edit protocols bgp family inet unicast]
local-ipv4-address local ipv4 address;
```

NOTE: You cannot configure this feature for the inet6 unicast, inet6 multicast, or inet6 labeled-unicast address families because BGP already has the capability to advertise these address families over an IPv6 BGP session.

The configured `local-ipv4-address` is used only when BGP advertises routes with self-next hop. When IBGP advertises routes learned from EBGP peers or the route reflector advertises BGP routes to its clients, BGP does not change the route next hop, ignores the configured `local-ipv4-address`, and uses the original IPv4 next hop.

SEE ALSO

local-ipv4-address

Example: Advertising IPv4 Routes over IPv6 BGP Sessions

IN THIS SECTION

- Requirements | **1096**
- Overview | **1096**
- Configuration | **1097**
- Verification | **1102**

This example shows how to advertise IPv4 routes over IPv6 BGP session.In a dual-stack environment that has IPv6 in its core, there is a need to reach remote IPv4 hosts. Therefore, BGP advertises IPv4 routes with IPv4 next hops to BGP peers over BGP sessions using IPv6 source and destination addresses. This feature enables BGP to advertise IPv4 unicast reachability with IPv4 next hop over IPv6 BGP sessions.

Requirements

This example uses the following hardware and software components:

- Three routers with dual stacking capability
- Junos OS Release 16.1 or later running on all the devices

Before you enable IPv4 advertisements over IPv6 BGP sessions, be sure to:

1. Configure the device interfaces.
2. Configure dual stacking on all devices.

Overview

IN THIS SECTION

- Topology | **1096**

Beginning with Release 16.1, Junos OS allows BGP to advertise IPv4 unicast reachability with IPv4 next hop over an IPv6 BGP session. In earlier Junos OS releases, BGP could advertise only inet6 unicast, inet6 multicast and inet6 labeled unicast address families over IPv6 BGP sessions. This feature allows BGP to exchange all BGP address families over an IPv6 session. You can enable BGP to advertise IPv4 routes with IPv4 next hops to BGP peers over IPv6 session. The configured `local-ipv4-address` is used only when BGP advertises routes with self-next hop.

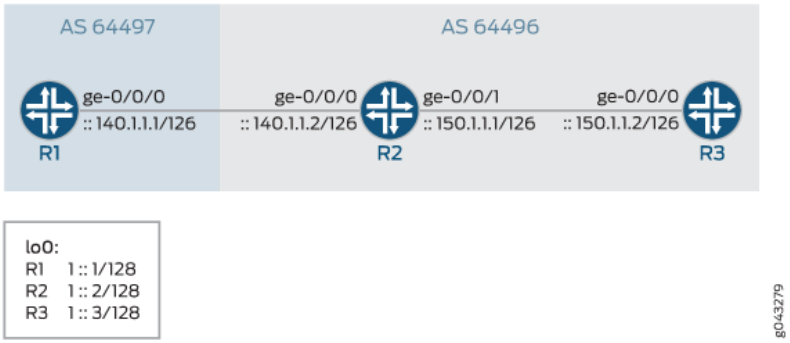
NOTE: You cannot configure this feature for the inet6 unicast, inet6 multicast, or inet6 labeled-unicast address families because BGP already has the capability to advertise these address families over an IPv6 BGP session.

Topology

In Figure 70 on page 1097, an IPv6 external BGP session is running between Routers R1 and R2. An IPv6 IBGP session is established between Router R2 and Router R3. IPv4 static routes are redistributed

to the BGP on R1. To redistribute the IPv4 routes over the IPv6 BGP session, the new feature must be enabled on all routers at the `[edit protocols bgp address family]` hierarchy level.

Figure 70: Advertising IPv4 Routes over IPv6 BGP Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | **1097**
- Configuring Router R1 | **1099**
- Results | **1100**

CLI Quick Configuration

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

Router R1

```
set interfaces ge-0/0/0 unit 0 description R1->R2
set interfaces ge-0/0/0 unit 0 family inet address 140.1.1.1/24
set interfaces ge-0/0/0 unit 0 family inet6 address ::140.1.1.1/126
set interfaces lo0 unit 0 family inet6 address 1::1/128
set routing-options static route 11.1.1.1/32 discard
set routing-options static route 11.1.1.2/32 discard
```

```
set routing-options autonomous-system 64497
set protocols bgp group ebgp-v6 type external
set protocols bgp group ebgp-v6 export p1
set protocols bgp group ebgp-v6 peer-as 64496
set protocols bgp group ebgp-v6 neighbor ::140.1.1.2 description R2
set protocols bgp group ebgp-v6 neighbor ::140.1.1.2 family inet unicast local-ipv4-address
140.1.1.1
set policy-options policy-statement p1 from protocol static
set policy-options policy-statement p1 then accept
```

Router R2

```
set interfaces ge-0/0/0 unit 0 description R2->R1
set interfaces ge-0/0/0 unit 0 family inet address 140.1.1.2/24
set interfaces ge-0/0/0 unit 0 family inet6 address ::140.1.1.2/126
set interfaces ge-0/0/1 unit 0 description R2->R3
set interfaces ge-0/0/1 unit 0 family inet address 150.1.1.1/24
set interfaces ge-0/0/1 unit 0 family inet6 address ::150.1.1.1/126
set interfaces lo0 unit 0 family inet6 address 1::2/128
set routing-options autonomous-system 64496
set protocols bgp group ibgp-v6 type internal
set protocols bgp group ibgp-v6 export change-nh
set protocols bgp group ibgp-v6 neighbor ::150.1.1.2 description R3
set protocols bgp group ibgp-v6 neighbor ::150.1.1.2 family inet unicast local-ipv4-address
150.1.1.1
set protocols bgp group ebgp-v6 type external
set protocols bgp group ebgp-v6 peer-as 64497
set protocols bgp group ebgp-v6 neighbor ::140.1.1.1 description R1
set protocols bgp group ebgp-v6 neighbor ::140.1.1.1 family inet unicast local-ipv4-address
140.1.1.2
set policy-options policy-statement change-nh from protocol bgp
set policy-options policy-statement change-nh then next-hop self
set policy-options policy-statement change-nh then accept
```

Router R3

```
set interfaces ge-0/0/0 unit 0 description R3->R2
set interfaces ge-0/0/0 unit 0 family inet address 150.1.1.2/24
set interfaces ge-0/0/0 unit 0 family inet6 address ::150.1.1.2/126
set interfaces lo0 unit 0 family inet6 address 1::3/128
set routing-options autonomous-system 64496
```

```
set protocols bgp group ibgp-v6 type internal
set protocols bgp group ibgp-v6 neighbor ::150.1.1.1 description R2
set protocols bgp group ibgp-v6 neighbor ::150.1.1.1 family inet unicast local-ipv4-address
150.1.1.2
```

Configuring Router R1

Step-by-Step Procedure

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

To configure Router R1:

NOTE: Repeat this procedure for other routers after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv4 and IPv6 addresses.

```
[edit interfaces]
user@R1# set ge-0/0/0 unit 0 description R1->R2
user@R1# set ge-0/0/0 unit 0 family inet address 140.1.1.1/24
user@R1# set ge-0/0/0 unit 0 family inet6 address ::140.1.1.1/126
```

2. Configure the loopback address.

```
[edit interfaces]
user@R1# set lo0 unit 0 family inet6 address 1::1/128
```

3. Configure an IPv4 static route that needs to be advertised.

```
[edit routing-options]
user@R1# set static route 11.1.1.1/32 discard
user@R1# set static route 11.1.1.2/32 discard
```

4. Configure the autonomous system for BGP hosts.

```
[edit routing-options]
user@R1# set autonomous-system 64497
```

5. Configure EBGP on the external edge routers.

```
[edit protocols]
user@R1# set bgp group ebgp-v6 type external
user@R1# set bgp group ebgp-v6 peer-as 64496
user@R1# set bgp group ebgp-v6 neighbor ::140.1.1.2 description R2
```

6. Enable the feature to advertise IPv4 adddress 140.1.1.1 over BGP IPv6 sessions.

```
[edit protocols]
user@R1# set bgp group ebgp-v6 neighbor ::140.1.1.2 family inet unicast local-ipv4-address
140.1.1.1
```

7. Define a policy p1 to accept all static routes.

```
[edit policy-options]
user@R1# set policy-statement p1 from protocol static
user@R1# set policy-statement p1 then accept
```

8. Apply the policy p1 on EBGP group ebgp-v6.

```
[edit protocols]
user@R1# set bgp group ebgp-v6 export p1
```

Results

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

```
[edit]
user@R1# show interfaces
```

```
ge-0/0/0 {
    unit 0 {
        description R1->R2;
        family inet {
            address 140.1.1.1/24;
        }
        family inet6 {
            address ::140.1.1.1/126;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 1::1/128;
            }
        }
    }
}
```

```
[edit]
user@R1# show protocols
bgp {
    group ebgp-v6 {
        type external;
        export p1;
        peer-as 64496;
        neighbor ::140.1.1.2 {
            description R2;
            family inet {
                unicast {
                    local-ipv4-address 140.1.1.1;
                }
            }
        }
    }
}
```

```
[edit]
user@R1# show routing-options
static {
```

```
    route 11.1.1.1/32 discard;
    route 11.1.1.2/32 discard;
}
autonomous-system 64497;
```

```
[edit]
user@R1# show policy-options
policy-statement p1 {
    from {
        protocol static;
    }
    then accept;
}
```

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

```
user@R1# commit
```

Verification

IN THIS SECTION

- Verifying That the BGP Session Is Up | **1102**
- Verifying That the IPv4 address Is Being Advertised | **1103**
- Verifying That the BGP Neighbor Router R2 Receives the Advertised IPv4 Address | **1104**

Confirm that the configuration is working properly.

Verifying That the BGP Session Is Up

Purpose

Verify that BGP is running on the configured interfaces and that the BGP session is active for each neighbor address.

Action

From operational mode, run the **show bgp summary** command on Router R1.

```
user@R1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
::140.1.1.2             64496       4140       4158       0       0  1d 7:10:36
0/0/0/0              0/0/0/0
```

Meaning

The BGP session is up and running, and BGP peering is established.

Verifying That the IPv4 address Is Being Advertised

Purpose

Verify that the configured IPv4 address is being advertised by Router R1 to the configured BGP neighbors.

Action

From operational mode, run the **show route advertising-protocol bgp ::150.1.1.2** command on Router R1.

```
user@R1> show route advertising-protocol bgp ::150.1.1.2
inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
  Prefix          Nexthop           MED     Lclpref    AS path
* 11.1.1.1/32       Self                          64497        64497 I
* 11.1.1.2/32       Self                          64497        64497 I
```

Meaning

The IPv4 static route is being advertised to the BGP neighbor Router R2.

Verifying That the BGP Neighbor Router R2 Receives the Advertised IPv4 Address

Purpose

Verify that Router R2 receives the IPv4 address that Router R1 is advertising to the BGP neighbor over IPv6.

Action

```
user@R2> show route receive-protocol bgp ::140.1.1.1
inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
  Prefix          Nexthop           MED     Lclpref    AS path
* 11.1.1.1/32        140.1.1.1                              64497 I
* 11.1.1.2/32        140.1.1.1                              64497 I

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 9 destinations, 10 routes (9 active, 0 holddown, 0 hidden)
```

Meaning

The presence of the static IPv4 route in Router R2's routing table indicates that it is receiving the advertised IPv4 routes from Router R1.

SEE ALSO

local-ipv4-address

Advertising IPv4 Routes over BGP IPv6 Sessions Overview | 1094

Understanding Redistribution of IPv4 Routes with IPv6 Next Hop into BGP

IN THIS SECTION

- BGP Next Hop Encoding | 1105

- Tunnel Localization | **1106**
- Tunnel Handling | **1106**
- Tunnel Load Balancing and Anchor Packet Forwarding Engine Failure Handling | **1110**
- Tunnel Loopback Stream Statistics | **1111**

In a network that predominantly transports IPv6 traffic there is a need to route IPv4 routes when required. For example, an Internet Service Provider that has an IPv6-only network, but has customers who still route IPv4 traffic. In this case, it is necessary to cater to such customers and forward IPv4 traffic over an IPv6 network. As described in RFC 5549, *Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop* IPv4 traffic is tunneled from customer premises equipment (CPE) devices to IPv4-over-IPv6 gateways. These gateways are announced to CPE devices through anycast addresses. The gateway devices then create dynamic IPv4-over-IPv6 tunnels to remote CPE devices and advertise IPv4 aggregate routes to steer traffic.

NOTE: Dynamic IPv4-over-IPv6 tunnel feature does not support unified ISSU in Junos OS Release 17.3R1.

Route reflectors (RRs) with a programmable interface are connected through IBGP to the gateway routers and host routes with IPv6 address as the next hop. These RRs advertise the IPv4 /32 addresses to inject the tunnel information into the network. The gateway routers create dynamic IPv4-over-IPv6 tunnels to the remote customer provider edge. The gateway router also advertises the IPv4 aggregate routes to steer traffic. The RR then advertises the tunnel source routes to the ISP. When the RR removes the tunnel route, BGP also withdraws the route causing the tunnel to be torn down and the CPE to be unreachable. The gateway router also withdraws the IPv4 aggregate routes and IPv6 tunnel source routes when all the aggregate routes contributor routes are removed. The gateway router sends route withdraw when the anchor Packet Forwarding Engine line card goes down, so that it will redirect traffic to other gateway routers.

The following extensions are introduced to support IPv4 routes with an IPv6 next hop:

BGP Next Hop Encoding

BGP is extended with next hop encoding capability that is used to send IPv4 routes with IPv6 next hops. If this capability is not available on the remote peer, BGP groups the peers based on this encoding capability and removes BGP family without encoding capability from the negotiated network layer reachability information (NLRI) list. Junos OS allows only one resolution table such as inet.0. To permit IPv4 BGP routes with IPv6 next hops BGP creates a new resolution tree. This feature allows a Junos OS routing table to have multiple resolution trees.

Besides RFC 5549, *Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop* a new encapsulation community specified in RFC 5512, *The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute* is introduced to determine the address family of the next-hop address. The encapsulation community indicates the type of tunnels that the ingress node needs to create. When BGP receives IPv4 routes with IPv6 next hop address and the V4oV6 encapsulation community, then BGP creates IPv4-over-IPv6 dynamic tunnels. When BGP receives routes without the encapsulation community, BGP routes are resolved without creating the V4oV6 tunnel.

A new policy action `dynamic-tunnel-attributes` *`dyan-attribute`* is available at the `[edit policy-statement` *`policy name`* `term then]` hierarchy level to support the new extended encapsulation.

Tunnel Localization

The dynamic tunnel infrastructure is enhanced with tunnel localization to support a larger number of tunnels. There is a need for tunnel localization to provide resiliency to handle traffic when the anchor fails. One or more chassis back up one another and let the routing protocol process (rpd) steer traffic away from the failure point to the backup chassis. The chassis advertises only these aggregate prefixes instead of the individual loopback addresses into the network.

Tunnel Handling

IPv4 over IPv6 tunnels use the dynamic tunnel infrastructure along with tunnel anchoring to support the required chassis wide scale. The tunnel state is localized to a Packet Forwarding Engine and the other Packet Forwarding Engines steer the traffic to the tunnel anchor.

Tunnel Ingress

Tunnel ingress or tunnel encapsulation forwards the network traffic towards the customer site. When the tunnel state is present on the Packet Forwarding Engine on which traffic entered the chassis, the routing protocol process (rpd) uses the following procedure to redistribute IPv4 routes over IPv6 tunnels:

Figure 71: Tunnel Ingress Handling when the Tunnel State is Available on the same PFE

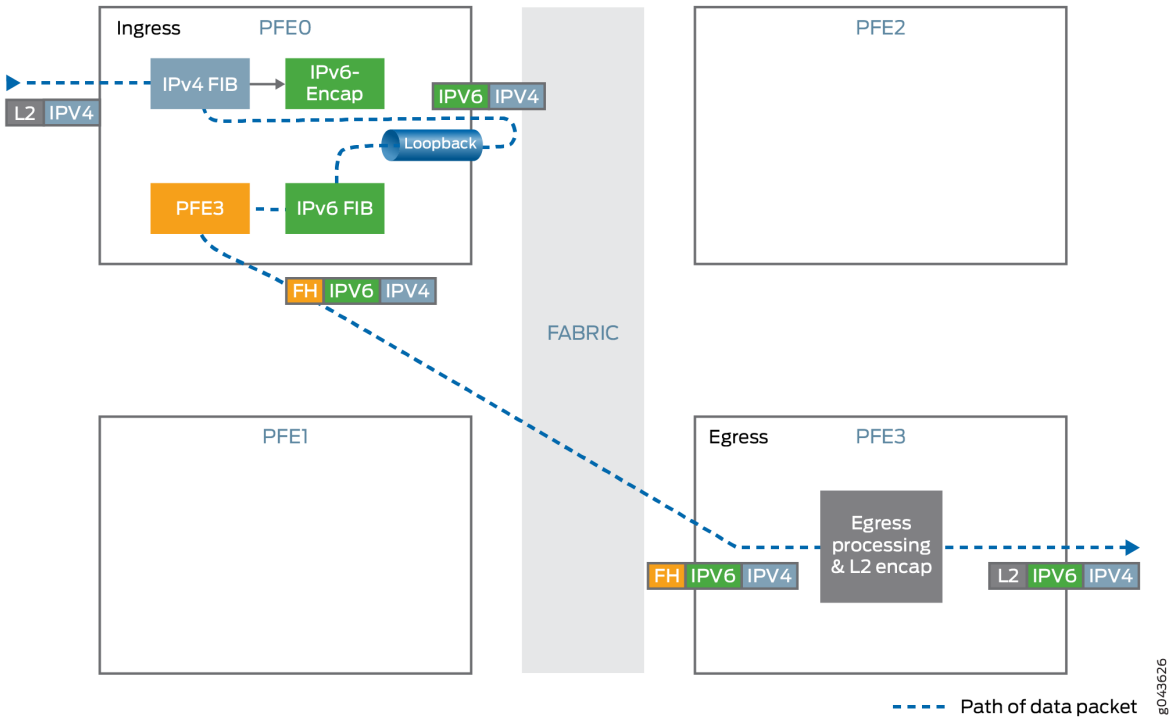
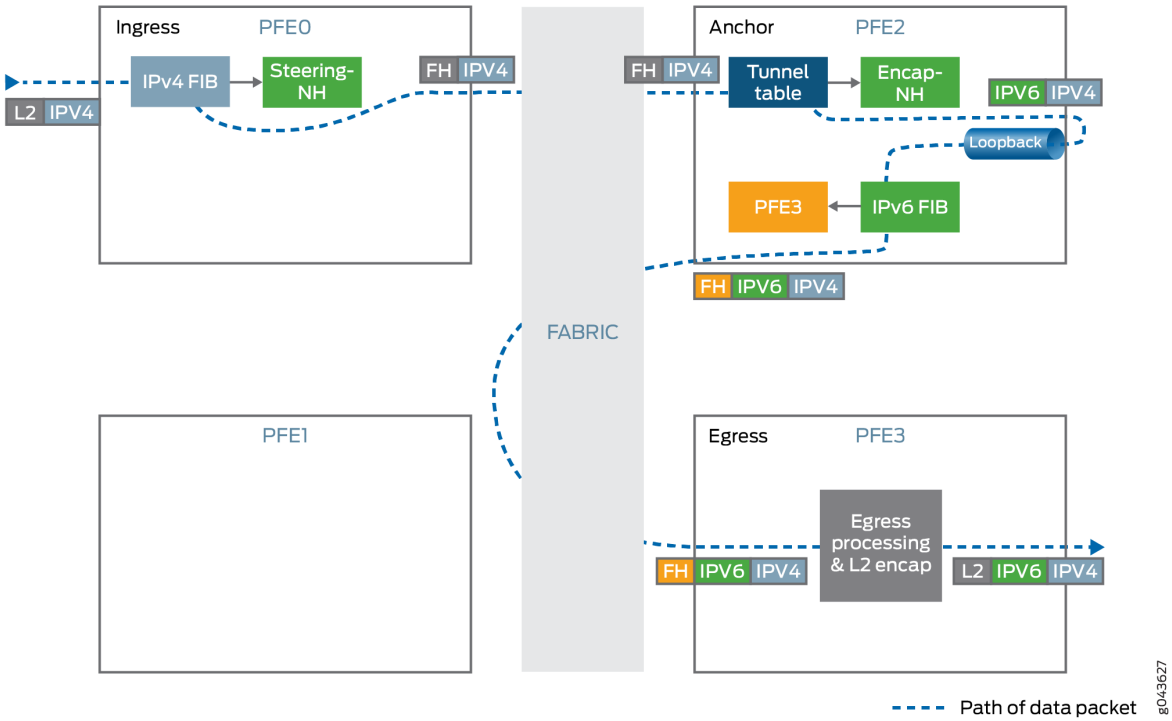


Figure 72: Tunnel Ingress Handling when the Tunnel State is on a Different PFE



1. Encapsulates IPv4 traffic inside the IPv6 header.

   Maximum transmission unit (MTU) enforcement is performed before encapsulation. If the encapsulated packet size exceeds the tunnel MTU and the IPv4 packet's `DF-bit` is not set then the packet is fragmented and these fragments are encapsulated.

2. Uses hash-based traffic load balancing on inner packet headers.

3. Forwards traffic to the destination IPv6 address. The IPv6 address is taken from the IPv6 header.

Tunnel Egress

Tunnel egress forwards traffic from the customer premises equipment to the network side.

Figure 73: Tunnel Egress Handling when the Tunnel State is Available on the same PFE

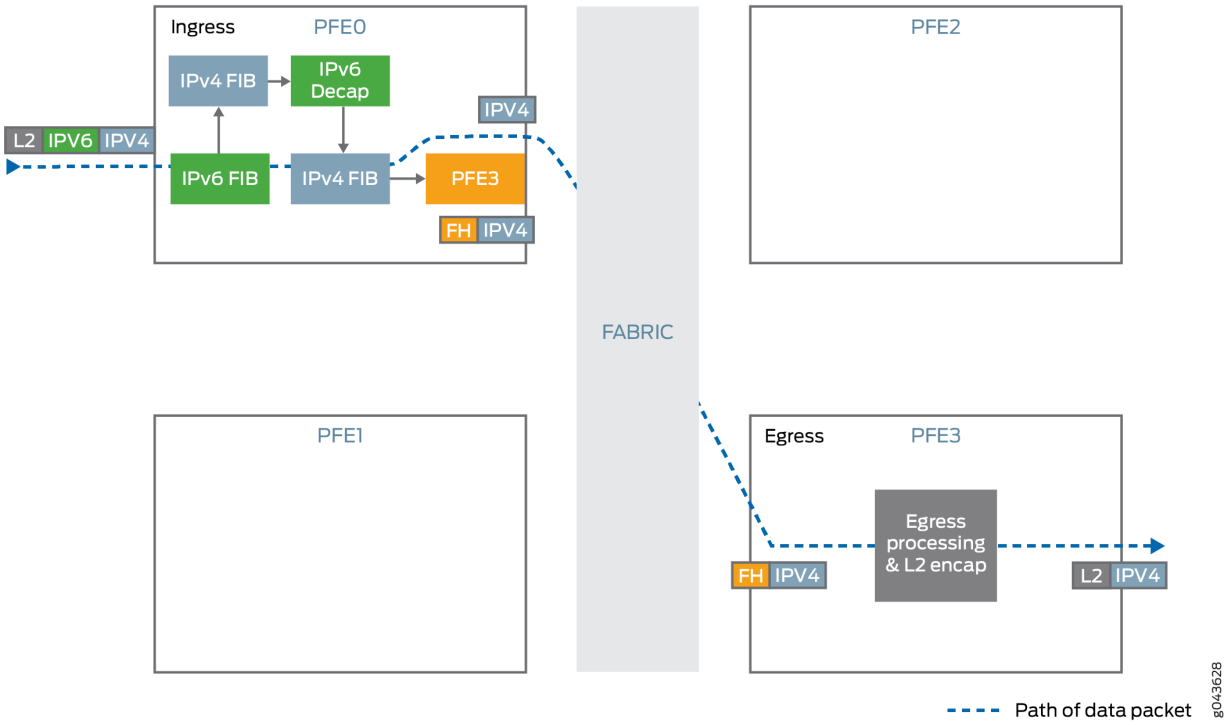
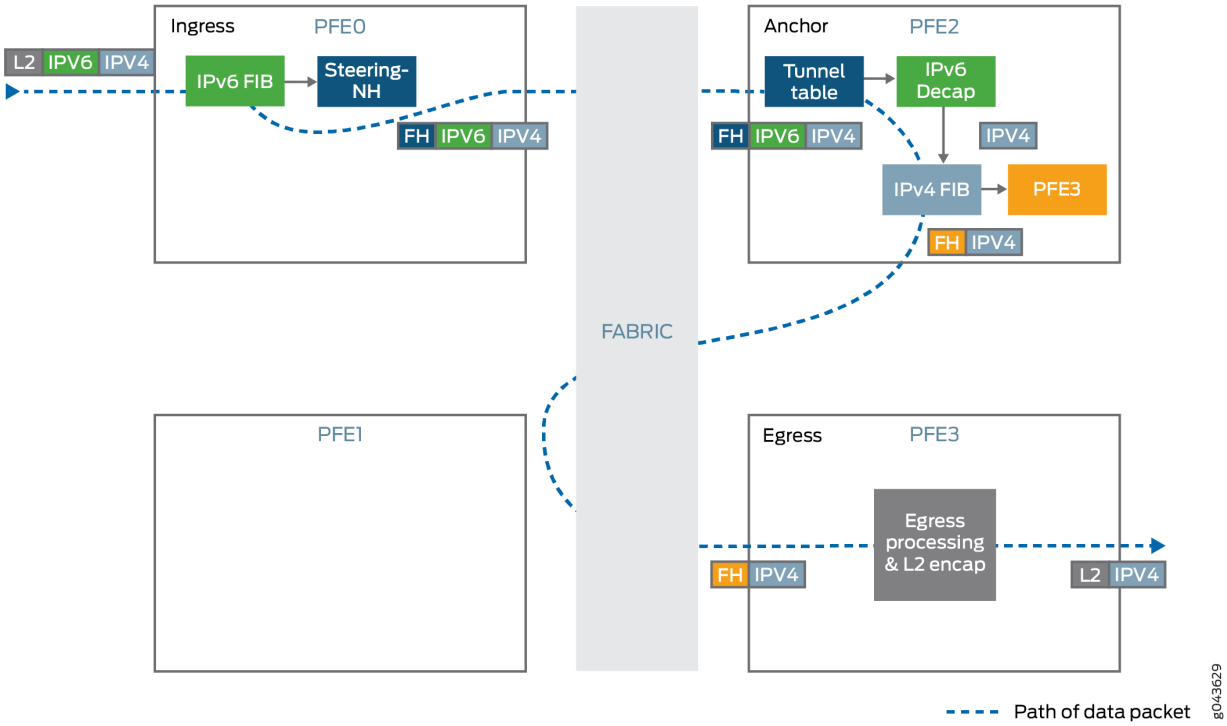


Figure 74: Tunnel Egress Handling when the Tunnel State is Available on a Remote PFE



1. Decapsulates the IPv4 packet present inside the IPv6 packet.
2. Performs anti-spoof checking to ensure that the IPv6, IPv4 pair matches with the information that was used for setting up the tunnel.
3. Looks up the IPv4 destination address from the decapsulated packet's IPv4 header and forwards the packet to the specified IPv4 address.

Tunnel Load Balancing and Anchor Packet Forwarding Engine Failure Handling

The Packet Forwarding Engine failure needs to be handled promptly to avoid null-route filtering of tunnel traffic anchored on the Packet Forwarding Engine. Tunnel localization involves the use of BGP advertisements to repair the failure globally. The tunnel traffic is diverted away from the failure point to other backup chassis that contains the identical tunnel state. For traffic load balancing, the chassis is configured to advertise different multiple exit discriminator (MED) values for each of the prefix sets so that only the traffic for one fourth of the tunnels goes through each chassis. CPE traffic is also handled in a similar manner by configuring the same set of anycast addresses on each chassis and steering only one fourth of traffic towards each chassis.

Anchor Packet Forwarding Engine is the single entity that does all processing for a tunnel. The anchor Packet Forwarding Engine selection is through static provisioning and tied to the Packet Forwarding Engine physical interfaces. When one of the Packet Forwarding Engines goes down, the daemon marks

all the Packet Forwarding Engines down on the line card and communicates this information to routing protocol process routing protocol process and other daemons. The routing protocol process sends out BGP withdrawals for the prefixes that are anchored on the failed Packet Forwarding Engine and the IPv6 addresses assigned to the Packet Forwarding Engine that is down. These advertisements reroute traffic to other backup chassis. When the failed Packet Forwarding Engine is up again, the chassis marks the Packet Forwarding Engine as `up` and updates routing protocol process. The routing protocol process triggers BGP updates to its peers that tunnels anchored to the specific Packet Forwarding Engine are now available for routing traffic. This process might take minutes for large scale tunnel configuration. Therefore, the `Ack` mechanism is built into the system to ensure minimal traffic loss while switching traffic back to the original chassis.

Tunnel Loopback Stream Statistics

Dynamic tunnel infrastructure uses loopback streams in Packet Forwarding Engine for looping the packet after encapsulation. Since the bandwidth of this loopback stream is limited there is a need to monitor the performance of tunnel loopback streams.

To monitor the statistics of the loopback stream, use the operational command `show pfe statistics traffic detail` that displays the aggregated loopback stream statistics including forwarding rate, drop packet rate and the byte rate.

SEE ALSO

- *dynamic-tunnels*
- *extended-nexthop*
- *tunnel-attributes*

Configuring BGP to Redistribute IPv4 Routes with IPv6 Next-Hop Addresses

Starting in Release 17.3R1, Junos OS devices can forward IPv4 traffic over an IPv6-only network, which generally cannot forward IPv4 traffic. As described in RFC 5549, IPv4 traffic is tunneled from CPE devices to IPv4-over-IPv6 gateways. These gateways are announced to CPE devices through anycast addresses. The gateway devices then create dynamic IPv4-over-IPv6 tunnels to remote customer premises equipment and advertise IPv4 aggregate routes to steer traffic. Route reflectors with programmable interfaces inject the tunnel information into the network. The route reflectors are connected through IBGP to gateway routers, which advertise the IPv4 addresses of host routes with IPv6 addresses as the next hop.

NOTE: Dynamic IPv4-over-IPv6 tunnel feature does not support unified ISSU in Junos OS Release 17.3R1.

Before you begin configuring BGP to distribute IPv4 routes with IPv6 next-hop addresses, do the following:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and LDP.
4. Configure BGP.

To configure BGP to distribute IPv4 routes with IPv6 next-hop addresses:

1. Configure the extended next-hop encoding option for BGP groups with IPv6 peers to route IPv4 address families over an IPv6 session.

```
[edit protocols bgp family inet unicast]
user@host# set extended-nexthop
```

2. Configure dynamic IPv4-over-IPv6 tunnels and define their attributes to forward IPv4 traffic over an IPv6-only network. IPv4 traffic is tunneled from CPE devices to IPv4-over-IPv6 gateways.

```
[edit routing-options]
user@host# set dynamic-tunnels
```

3. Configure the tunnel attributes.

```
[edit routing-options dynamic-tunnels tunnel-attributes]
user@host# set tunnel-attributes name
user@host# set dynamic-tunnel-source-prefix dynamic-tunnel-source-prefix
user@host# set dynamic-tunnel-type V4oV6
user@host# set dynamic-tunnel-mtu dynamic-tunnel-mtu
user@host# set dynamic-tunnel-anchor-pfe dynamic-tunnel-anchor-pfe
user@host# set dynamic-tunnel-anti-spoof (off | on)
```

For example, configure a dynamic tunnel, `first_tunnel` with the following attributes:

```
[edit routing-options dynamic-tunnels tunnel-attributes]
user@host# set tunnel-attributes first_tunnel
user@host# set dynamic-tunnel-source-prefix 10.1.1.0
user@host# set dynamic-tunnel-type V4oV6
user@host# set dynamic-tunnel-mtu 300
user@host# set dynamic-tunnel-anchor-pfe pfe-1/2/0
user@host# set dynamic-tunnel-anti-spoof on
```

4. Define a policy to associate the configured dynamic tunnel attribute profile to a prefix list or a route filter.

```
[edit policy-options policy-statement policy-name from then]
user@host# set dynamic-tunnel-attributes name
```

For example, define *dynamic_tunnel_policy* policy to associate the dynamic tunnel *first_tunnel* attributes only to traffic heading to a specific route 2.2.2.2/32.

```
[edit policy-options policy-statement dynamic_tunnel_policy from route-filter 2.2.2.2/32
exact then]
user@host# set dynamic-tunnel-attributes first_tunnel
```

5. Export the defined policy.

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

For example, export the configured *dynamic_tunnel_policy* policy.

```
[edit routing options]
user@host# set forwarding-table export dynamic_tunnel_policy
```

SEE ALSO

dynamic-tunnels

extended-nexthop

tunnel-attributes

Understanding Redistribution of IPv4 Routes with IPv6 Next Hop into BGP | 1104

Enabling Layer 2 VPN and VPLS Signaling

You can enable BGP to carry Layer 2 VPN and VPLS NLRI messages.

To enable VPN and VPLS signaling, include the `family` statement:

```
family {
    l2vpn {
        signaling {
            prefix-limit {
                maximum number;
                teardown <percentage> <idle-timeout (forever | minutes)>;
                drop-excess <percentage>;
                hide-excess <percentage>;            }
        }
    }
}
```

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

To configure a maximum number of prefixes, include the `prefix-limit` statement:

```
prefix-limit {
    maximum number;
    teardown <percentage> <idle-timeout (forever | minutes)>;
    drop-excess <percentage>;
    hide-excess <percentage>;}
```

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

When you set the maximum number of prefixes, a message is logged when that number is reached. If you include the `teardown` statement, the session is torn down when the maximum number of prefixes is reached. If you specify a percentage, messages are logged when the number of prefixes reaches that percentage. Once the session is torn down, it is reestablished in a short time. Include the `idle-timeout` statement to keep the session down for a specified amount of time, or forever. If you specify `forever`, the

session is reestablished only after you use the `clear bgp neighbor` command. If you include the `drop-excess <percentage>` statement and specify a percentage, the excess routes are dropped when the number of prefixes exceeds the percentage. If you include the `hide-excess <percentage>` statement and specify a percentage, the excess routes are hidden when the number of prefixes exceeds the percentage. If the percentage is modified, the routes are re-evaluated automatically.

SEE ALSO

Junos OS VPNs Library for Routing Devices

Understanding BGP Flow Routes for Traffic Filtering

IN THIS SECTION

- Match Conditions for Flow Routes | **1116**
- Actions for Flow Routes | **1120**
- Validating Flow Routes | **1121**
- Support for BGP Flow-Specification Algorithm Version 7 and Later | **1121**

A flow route is an aggregation of match conditions for IP packets. Flow routes are installed as Input Forwarding Table Filters (implicit) and are propagated through the network using flow-specification network-layer reachability information (NLRI) messages and installed into the flow routing table *instance-name*.`inetflow.0`. Packets can travel through flow routes only if specific match conditions are met.

Flow routes and firewall filters are similar in that they filter packets based on their components and perform an action on the packets that match. Flow routes provide traffic filtering and rate-limiting capabilities much like firewall filters. In addition, you can propagate flow routes across different autonomous systems.

Flow routes are propagated by BGP through flow-specification NLRI messages. You must enable BGP to propagate these NLRIs.

Beginning with Junos OS Release 15.1, changes are implemented to extend nonstop active routing (NSR) support for existing inet-flow and inetvpn-flow families and extend route validation for BGP flowspec per draft-ietf-idr-bgp-flowspec-oid-01. Two new statements are introduced as part of this enhancement. See *enforce-first-as* and *no-install*.

NOTE: Beginning with Junos OS Release 16.1, IPv6 support is extended to BGP flow specification that allows propagation of traffic flow specification rules for IPv6 and VPN-IPv6 packets. BGP flow specification automates coordination of traffic filtering rules in order to mitigate distributed denial-of-service attack during nonstop active routing (NSR).

Starting with Junos OS Release 16.1R1, BGP flow specification supports traffic-marking `extended-community` filtering action. For IPv4 traffic, Junos OS modifies the DiffServ code point (DSCP) bits of a transiting IPv4 packet to the corresponding value of the extended community. For IPv6 packets, Junos OS modifies the first six bits of the `traffic class` field of the transmitting IPv6 packet to the corresponding value of the extended community.

Starting in Junos OS Release 17.1R1, BGP can carry flow-specification network layer reachability information (NLRI) messages on PTX Series routers that have third-generation FPCs (FPC3-PTX-U2 and FPC3-PTX-U3 on PTX5000 and FPC3-SFF-PTX-U0 and FPC3-SFF-PTX-U1 on PTX3000) installed. Propagating firewall filter information as part of BGP enables you to propagate firewall filters against denial-of-service (DOS) attacks dynamically across autonomous systems.

Starting in Junos OS Release 17.2R1, BGP can carry flow-specification network layer reachability information (NLRI) messages on PTX1000 routers that have third-generation FPCs installed. Propagating firewall filter information as part of BGP enables you to propagate firewall filters against denial-of-service (DOS) attacks dynamically across autonomous systems.

Starting in cRPD Release 20.3R1, flow routes and policing rules propagated through BGP flow specification NLRI are downloaded to Linux kernel through Linux Netfilter framework on cRPD environments.

Match Conditions for Flow Routes

You specify conditions that the packet must match before the action in the `then` statement is taken for a flow route. All conditions in the `from` statement must match for the action to be taken. The order in which you specify match conditions is not important, because a packet must match all the conditions in a term for a match to occur.

To configure a match condition, include the `match` statement at the `[edit routing-options flow]` hierarchy level.

Table 8 on page 1117 describes the flow route match conditions.

Table 8: Flow Route Match Conditions

Match Condition	Description
`destination` *prefix* `prefix-offset` *number*	IP destination address field. You can use the `prefix-offset` optional field, which is available only on Junos devices with enhanced MPCs that are configured for `enhanced-ip` mode, to specify the number of bits that must be skipped before Junos OS starts matching an IPv6 prefix.
`destination-port` *number*	TCP or User Datagram Protocol (UDP) destination port field. You cannot specify both the `port` and `destination-port` match conditions in the same term. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): `afs` (1483), `bgp` (179), `biff` (512), `bootpc` (68), `bootps` (67), `cmd` (514), `cvspserver` (2401), `dhcp` (67), `domain` (53), `eklogin` (2105), `ekshell` (2106), `exec` (512), `finger` (79), `ftp` (21), `ftp-data` (20), `http` (80), `https` (443), `ident` (113), `imap` (143), `kerberos-sec` (88), `klogin` (543), `kpasswd` (761), `krb-prop` (754), `krbupdate` (760), `kshell` (544), `ldap` (389), `login` (513), `mobileip-agent` (434), `mobilip-mn` (435), `msdp` (639), `netbios-dgm` (138), `netbios-ns` (137), `netbios-ssn` (139), `nfsd` (2049), `nntp` (119), `ntalk` (518), `ntp` (123), `pop3` (110), `pptp` (1723), `printer` (515), `radacct` (1813), `radius` (1812), `rip` (520), `rkinit` (2108), `smtp` (25), `snmp` (161), `snmptrap` (162), `snpp` (444), `socks` (1080), `ssh` (22), `sunrpc` (111), `syslog` (514), `tacacs-ds` (65), `talk` (517), `telnet` (23), `tftp` (69), `timed` (525), `who` (513), `xdmcp` (177), `zephyr-clt` (2103), or `zephyr-hm` (2104).
`dscp` *number*	Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant six bits of this byte form the DSCP. You can specify DSCP in hexadecimal or decimal form.
`flow-label` *numeric-expression*	Match the flow label value. The value of this field ranges from 0 through 1048575. This match condition is supported only on Junos devices with enhanced MPCs that are configured for `enhanced-ip` mode. This match condition is not supported for IPv4.

Table 8: Flow Route Match Conditions *(Continued)*

Match Condition	Description
`fragment` *type*	Fragment type field. The keywords are grouped by the fragment type with which they are associated: • `dont-fragment` **NOTE**: This option is not supported for IPv6. • `first-fragment` • `is-fragment` • `last-fragment` • `not-a-fragment` This match condition is supported only on Junos OS devices with enhanced MPCs that are configured for `enhanced-ip` mode. .
`icmp-code` *number*`icmp6-code` *icmp6-code-value*;	ICMP code field. This value or keyword provides more specific information than `icmp-type`. Because the value's meaning depends upon the associated `icmp-type` value, you must specify `icmp-type` along with `icmp-code`. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: `ip-header-bad` (0), `required-option-missing` (1) • redirect: `redirect-for-host` (1), `redirect-for-network` (0), `redirect-for-tos-and-host` (3), `redirect-for-tos-and-net` (2) • time-exceeded: `ttl-eq-zero-during-reassembly` (1), `ttl-eq-zero-during-transit` (0) • unreachable: `communication-prohibited-by-filtering` (13), `destination-host-prohibited` (10), `destination-host-unknown` (7), `destination-network-prohibited` (9), `destination-network-unknown` (6), `fragmentation-needed` (4), `host-precedence-violation` (14), `host-unreachable` (1), `host-unreachable-for-TOS` (12), `network-unreachable` (0), `network-unreachable-for-TOS` (11), `port-unreachable` (3), `precedence-cutoff-in-effect` (15), `protocol-unreachable` (2), `source-host-isolated` (8), `source-route-failed` (5)

Table 8: Flow Route Match Conditions *(Continued)*

Match Condition	Description
`icmp-type` *number* `icmp6-type` *icmp6-type-value*	ICMP packet type field. Normally, you specify this match in conjunction with the `protocol` match statement to determine which protocol is being used on the port. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): `echo-reply` (0), `echo-request` (8), `info-reply` (16), `info-request` (15), `mask-request` (17), `mask-reply` (18), `parameter-problem` (12), `redirect` (5), `router-advertisement` (9), `router-solicit` (10), `source-quench` (4), `time-exceeded` (11), `timestamp` (13), `timestamp-reply` (14), or `unreachable` (3).
`packet-length` *number*	Total IP packet length.
`port` *number*	TCP or UDP source or destination port field. You cannot specify both the `port` match and either the `destination-port` or `source-port` match condition in the same term. In place of the numeric value, you can specify one of the text synonyms listed under `destination-port`.
`protocol` *number*	IP protocol field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): `ah`, `egp` (8), `esp` (50), `gre` (47), `icmp` (1), `igmp` (2), `ipip` (4), `ipv6` (41), `ospf` (89), `pim` (103), `rsvp` (46), `tcp` (6), or `udp` (17). This match condition is supported for IPv6 only on Junos devices with enhanced MPCs that are configured for `enhanced-ip` mode.
`source` *prefix*`prefix-offset` *number*	IP source address field. You can use the `prefix-offset` optional field, which is available only on Junos devices with enhanced MPCs that are configured for `enhanced-ip` mode, to specify the number of bits that must be skipped before Junos OS starts matching an IPv6 prefix.
`source-port` *number*	TCP or UDP source port field. You cannot specify the `port` and `source-port` match conditions in the same term. In place of the numeric field, you can specify one of the text synonyms listed under `destination-port`.

Table 8: Flow Route Match Conditions *(Continued)*

Match Condition	Description
`tcp-flag type`	TCP header format.

Actions for Flow Routes

You can specify the action to take if the packet matches the conditions you have configured in the flow route. To configure an action, include the `then` statement at the `[edit routing-options flow]` hierarchy level.

Table 9 on page 1120 describes the flow route actions.

Table 9: Flow Route Action Modifiers

Action or Action Modifier	Description
Actions	
`accept`	Accept a packet. This is the default.
`discard`	Discard a packet silently, without sending an Internet Control Message Protocol (ICMP) message.
`community`	Replace any communities in the route with the specified communities.
mark *value*	Set a DSCP value for traffic that matches this flow. Specify a value from 0 through 63. This action is supported only on Junos devices with enhanced MPCs that are configured for `enhanced-ip` mode.
`next term`	Continue to the next match condition for evaluation.
`routing-instance` *`extended-community`*	Specify a routing instance to which packets are forwarded.
`rate-limit` *`bits-per-second`*	Limit the bandwidth on the flow route. Express the limit in bits per second (bps). Beginning with Junos OS Release 16.1R4, the rate-limit range is [0 through 1000000000000].

Table 9: Flow Route Action Modifiers *(Continued)*

Action or Action Modifier	Description
`sample`	Sample the traffic on the flow route.

Validating Flow Routes

The Junos OS installs flow routes into the flow routing table only if they have been validated using the validation procedure. The Routing Engine does the validation before the installing routes into the flow routing table.

Flow routes received using the BGP network layer reachability information (NLRI) messages are validated before they are installed into the flow primary instance routing table `instance.inetflow.0`. The validation procedure is described in the draft-ietf-idr-flow-spec-09.txt, *Dissemination of Flow Specification Rules*. You can bypass the validation process for flow routes using BGP NLRI messages and use your own specific import policy.

To trace validation operations, include the `validation` statement at the `[edit routing-options flow]` hierarchy level.

Support for BGP Flow-Specification Algorithm Version 7 and Later

By default, the Junos OS uses the term-ordering algorithm defined in version 6 of the BGP flow specification draft. In Junos OS Release 10.0 and later, you can configure the router to comply with the term-ordering algorithm first defined in version 7 of the BGP flow specification and supported through RFC 5575, *Dissemination of Flow Specification Routes*.

BEST PRACTICE: We recommend that you configure the Junos OS to use the term-ordering algorithm first defined in version 7 of the BGP flow specification draft. We also recommend that you configure the Junos OS to use the same term-ordering algorithm on all routing instances configured on a router.

To configure BGP to use the flow-specification algorithm first defined in version 7 of the Internet draft, include the `standard` statement at the `[edit routing-options flow term-order]` hierarchy level.

To revert to using the term-ordering algorithm defined in version 6, include the `legacy` statement at the `[edit routing-options flow term-order]` hierarchy level.

NOTE: The configured term order has only local significance. That is, the term order does not propagate with flow routes sent to the remote BGP peers, whose term order is completely determined by their own term order configuration. Therefore, you should be careful when configuring the order-dependent action `next term` when you are not aware of the term order configuration of the remote peers. The local `next term` might differ from the `next term` configured on the remote peer.

NOTE: On Junos OS Evolved, `next term` cannot appear as the last term of the action. A filter term where `next term` is specified as an action but without any match conditions configured is not supported.

Starting in Junos OS Release 16.1, you have the option to not apply the **flowspec** filter to traffic received on specific interfaces. A new term is added at the beginning of the **flowspec** filter that accepts any packet received on these specific interfaces. The new term is a variable that creates an exclusion list of terms attached to the forwarding table filter as a part of the flow specification filter.

To exclude the **flowspec** filter from being applied to traffic received on specific interfaces, you must first configure a `group-id` on such interfaces by including the family `inet` filter group `group-id` statement at the `[edit interfaces]` hierarchy level and then attach the **flowspec** filter with the interface group by including the `flow interface-group group-id exclude` statement at the `[edit routing-options]` hierarchy level. You can configure only one `group-id` per routing instance with the `set routing-options flow interface-group group-id` statement.

SEE ALSO

interface-group

flow (IPv6)

group

flow

Example: Enabling BGP to Carry Flow-Specification Routes

IN THIS SECTION

- Requirements | 1123
- Overview | 1123
- Configuration | 1126
- Verification | 1137

This example shows how to allow BGP to carry flow-specification network layer reachability information (NLRI) messages.

Requirements

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol (IGP).
- Configure BGP.
- Configure a routing policy that exports routes (such as direct routes or IGP routes) from the routing table into BGP.

Overview

IN THIS SECTION

- Topology | 1125

Propagating firewall filter information as part of BGP enables you to propagate firewall filters against denial-of-service (DOS) attacks dynamically across autonomous systems. Flow routes are encapsulated into the flow-specification NLRI and propagated through a network or virtual private networks (VPNs), sharing filter-like information. Flow routes are an aggregation of match conditions and resulting actions for packets. They provide you with traffic filtering and rate-limiting capabilities much like firewall filters.

Unicast flow routes are supported for the default instance, VPN routing and forwarding (VRF) instances, and virtual-router instances.

Import and export policies can be applied to the family `inet flow` or family `inet-vpn flow` NLRI, affecting the flow routes accepted or advertised, similar to the way import and export policies are applied to other BGP families. The only difference is that the flow policy configuration must include the from `rib inetflow.0` statement. This statement causes the policy to be applied to the flow routes. An exception to this rule occurs if the policy has only the `then reject` or the `then accept` statement and no `from` statement. Then, the policy affects all routes, including IP unicast and IP flow.

The flow route filters are first configured on a router statically, with a set of matching criteria followed by the actions to be taken. Then, in addition to `family inet unicast`, `family inet flow` (or `family inet-vpn flow`) is configured between this BGP-enabled device and its peers.

By default, statically configured flow routes (firewall filters) are advertised to other BGP-enabled devices that support the `family inet flow` or `family inet-vpn flow` NLRI.

The receiving BGP-enabled device performs a validation process before installing the firewall filter into the flow routing table *`instance-name`*`.inetflow.0`. The validation procedure is described in RFC 5575, *Dissemination of Flow Specification Rules*.

The receiving BGP-enabled device accepts a flow route if it passes the following criteria:

- The originator of a flow route matches the originator of the best match unicast route for the destination address that is embedded in the route.
- There are no more specific unicast routes, when compared to the destination address of the flow route, for which the active route has been received from a different next-hop autonomous system.

The first criterion ensures that the filter is being advertised by the next-hop used by unicast forwarding for the destination address embedded in the flow route. For example, if a flow route is given as 10.1.1.1, proto=6, port=80, the receiving BGP-enabled device selects the more specific unicast route in the unicast routing table that matches the destination prefix 10.1.1.1/32. On a unicast routing table containing 10.1/16 and 10.1.1/24, the latter is chosen as the unicast route to compare against. Only the active unicast route entry is considered. This follows the concept that a flow route is valid if advertised by the originator of the best unicast route.

The second criterion addresses situations in which a given address block is allocated to different entities. Flows that resolve to a best-match unicast route that is an aggregate route are only accepted if they do not cover more specific routes that are being routed to different next-hop autonomous systems.

You can bypass the validation process for flow routes using BGP NLRI messages and use your own specific import policy. When BGP is carrying flow-specification NLRI messages, the `no-validate` statement at the `[edit protocols bgp group` *`group-name`* `family inet flow]` hierarchy level omits the flow route validation procedure after packets are accepted by a policy. You can configure the import policy to match on destination address and path attributes such as community, next-hop, and AS path. You can specify the

action to take if the packet matches the conditions you have configured in the flow route. To configure an action, include the statement at the `[edit routing-options flow]` hierarchy level. The flow specification NLRI type includes components such as destination prefix, source prefix, protocol, and ports as defined in the RFC 5575. The import policy can filter an inbound route using path attributes and destination address in the flow specification NLRI. The import policy cannot filter any other components in the RFC 5575.

The flow specification defines required protocol extensions to address most common applications of IPv4 unicast and VPN unicast filtering. The same mechanism can be reused and new match criteria added to address similar filtering for other BGP address families (for example, IPv6 unicast).

After a flow route is installed in the `inetflow.0` table, it is also added to the list of firewall filters in the kernel.

On routers only, flow-specification NLRI messages are supported in VPNs. The VPN compares the route target extended community in the NLRI to the import policy. If there is a match, the VPN can start using the flow routes to filter and rate-limit packet traffic. Received flow routes are installed into the flow routing table *instance-name*.`inetflow.0`. Flow routes can also be propagated throughout a VPN network and shared among VPNs. To enable multiprotocol BGP (MP-BGP) to carry flow-specification NLRI for the `inet-vpn` address family, include the `flow` statement at the `[edit protocols bgp group` *group-name* `family inet-vpn]` hierarchy level. VPN flow routes are supported for the default instance only. Flow routes configured for VPNs with family `inet-vpn` are not automatically validated, so the `no-validate` statement is not supported at the `[edit protocols bgp group` *group-name* `family inet-vpn]` hierarchy level. No validation is needed if the flow routes are configured locally between devices in a single AS.

Import and export policies can be applied to the `family inet flow` or `family inet-vpn flow` NLRI, affecting the flow routes accepted or advertised, similar to the way import and export policies are applied to other BGP families. The only difference is that the flow policy configuration must include the `from rib inetflow.0` statement. This statement causes the policy to be applied to the flow routes. An exception to this rule occurs if the policy has only the `then reject` or the `then accept` statement and no `from` statement. Then, the policy affects all routes, including IP unicast and IP flow.

This example shows how to configure the following export policies:

- A policy that allows the advertisement of flow routes specified by a route-filter. Only the flow routes covered by the 10.13/16 block are advertised. This policy does not affect unicast routes.
- A policy that allows all unicast and flow routes to be advertised to the neighbor.
- A policy that disallows all routes (unicast or flow) to be advertised to the neighbor.

Topology

Configuration

IN THIS SECTION

- Configuring a Static Flow Route | 1126
- Advertising Flow Routes Specified by a Route Filter | 1128
- Advertising All Unicast and Flow Routes | 1130
- Advertising No Unicast or Flow Routes | 1132
- Limiting the Number of Flow Routes Installed in a Routing Table | 1134
- Limiting the Number of Prefixes Received on a BGP Peering Session | 1136

Configuring a Static Flow Route

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.

```
set routing-options flow route block-10.131.1.1 match destination 10.131.1.1/32
set routing-options flow route block-10.131.1.1 match protocol icmp
set routing-options flow route block-10.131.1.1 match icmp-type echo-request
set routing-options flow route block-10.131.1.1 then discard
set routing-options flow term-order standard
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the match conditions.

```
[edit routing-options flow route block-10.131.1.1]
user@host# set match destination 10.131.1.1/32
```

```
user@host# set match protocol icmp
user@host# set match icmp-type echo-request
```

2. Configure the action.

```
[edit routing-options flow route block-10.131.1.1]
user@host# set then discard
```

3. (Recommended) For the flow specification algorithm, configure the standard-based term order.

```
[edit routing-options flow]
user@host# set term-order standard
```

In the default term ordering algorithm, as specified in the flowspec RFC draft Version 6, a term with less specific matching conditions is always evaluated before a term with more specific matching conditions. This causes the term with more specific matching conditions to never be evaluated. Version 7 of RFC 5575 made a revision to the algorithm so that the more specific matching conditions are evaluated before the less specific matching conditions. For backward compatibility, the default behavior is not altered in Junos OS, even though the newer algorithm makes more sense. To use the newer algorithm, include the `term-order standard` statement in the configuration. This statement is supported in Junos OS Release 10.0 and later.

Results

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

```
[edit]
user@host# show routing-options
flow {
    term-order standard;
    route block-10.131.1.1 {
        match {
            destination 10.131.1.1/32;
            protocol icmp;
            icmp-type echo-request;
        }
        then discard;
```

```
    }
}
```

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

Advertising Flow Routes Specified by a Route Filter

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.

```
set protocols bgp group core family inet unicast
set protocols bgp group core family inet flow
set protocols bgp group core export p1
set protocols bgp group core peer-as 65000
set protocols bgp group core neighbor 10.12.99.5
set policy-options policy-statement p1 term a from rib inetflow.0
set policy-options policy-statement p1 term a from route-filter 10.13.0.0/16 orlonger
set policy-options policy-statement p1 term a then accept
set policy-options policy-statement p1 term b then reject
set routing-options autonomous-system 65001
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the BGP group.

```
[edit protocols bgp group core]
user@host# set family inet unicast
user@host# set family inet flow
user@host# set export p1
user@host# set peer-as 65000
user@host# set neighbor 10.12.99.5
```

2. Configure the flow policy.

```
[edit policy-options policy-statement p1]
user@host# set term a from rib inetflow.0
user@host# set term a from route-filter 10.13.0.0/16 orlonger
user@host# set term a then accept
user@host# set term b then reject
```

3. Configure the local autonomous system (AS) number.

```
[edit routing-options]
user@host# set autonomous-system 65001
```

Results

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

```
[edit]
user@host# show protocols
bgp {
    group core {
        family inet {
            unicast;
            flow;
        }
        export p1;
        peer-as 65000;
        neighbor 10.12.99.5;
    }
}
```

```
[edit]
user@host# show policy-options
policy-statement p1 {
    term a {
        from {
```

```
            rib inetflow.0;
            route-filter 10.13.0.0/16 orlonger;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
```

```
[edit]
user@host# show routing-options
autonomous-system 65001;
```

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

Advertising All Unicast and Flow Routes

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.

```
set protocols bgp group core family inet unicast
set protocols bgp group core family inet flow
set protocols bgp group core export p1
set protocols bgp group core peer-as 65000
set protocols bgp group core neighbor 10.12.99.5
set policy-options policy-statement p1 term a then accept
set routing-options autonomous-system 65001
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the BGP group.

```
[edit protocols bgp group core]
user@host# set family inet unicast
user@host# set family inet flow
user@host# set export p1
user@host# set peer-as 65000
user@host# set neighbor 10.12.99.5
```

2. Configure the flow policy.

```
[edit policy-options policy-statement p1]
user@host# set term a then accept
```

3. Configure the local autonomous system (AS) number.

```
[edit routing-options]
user@host# set autonomous-system 65001
```

Results

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

```
[edit]
user@host# show protocols
bgp {
    group core {
        family inet {
            unicast;
            flow;
        }
        export p1;
        peer-as 65000;
        neighbor 10.12.99.5;
```

```
    }
}
```

```
[edit]
user@host# show policy-options
policy-statement p1 {
    term a {
        prefix-list inetflow;
        }
        then accept;
    }
}
```

```
[edit]
user@host# show routing-options
autonomous-system 65001;
```

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

Advertising No Unicast or Flow Routes

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.

```
set protocols bgp group core family inet unicast
set protocols bgp group core family inet flow
set protocols bgp group core export p1
set protocols bgp group core peer-as 65000
set protocols bgp group core neighbor 10.12.99.5
set policy-options policy-statement p1 term a then reject
set routing-options autonomous-system 65001
```

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 Junos OS CLI User Guide.

To configure the BGP peer sessions:

1. Configure the BGP group.

```
[edit protocols bgp group core]
user@host# set family inet unicast
user@host# set family inet flow
user@host# set export p1
user@host# set peer-as 65000
user@host# set neighbor 10.12.99.5
```

2. Configure the flow policy.

```
[edit policy-options policy-statement p1]
user@host# set term a then reject
```

3. Configure the local autonomous system (AS) number.

```
[edit routing-options]
user@host# set autonomous-system 65001
```

Results

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

```
[edit]
user@host# show protocols
bgp {
    group core {
        family inet {
            unicast;
```

```
            flow;
        }
        export p1;
        peer-as 65000;
        neighbor 10.12.99.5;
    }
}
```

```
[edit]
user@host# show policy-options
policy-statement p1 {
    term a {
        then reject;
    }
}
```

```
[edit]
user@host# show routing-options
autonomous-system 65001;
```

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

Limiting the Number of Flow Routes Installed in a Routing Table

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.

```
set routing-options rib inetflow.0 maximum-prefixes 1000
set routing-options rib inetflow.0 maximum-prefixes threshold 50
```

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 Junos OS CLI User Guide.

NOTE: Application of a route limit might result in unpredictable dynamic route protocol behavior. For example, once the limit is reached and routes are being rejected, BGP does not necessarily attempt to reinstall the rejected routes after the number of routes drops below the limit. BGP sessions might need to be cleared to resolve this issue.

To limit the flow routes:

1. Set an upper limit for the number of prefixes installed in `inetflow.0` table.

```
[edit routing-options rib inetflow.0]
user@host# set maximum-prefixes 1000
```

2. Set a threshold value of 50 percent, where when 500 routes are installed, a warning is logged in the system log.

```
[edit routing-options rib inetflow.0]
user@host# set maximum-prefixes threshold 50
```

Results

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

```
[edit]
user@host# show routing-options
rib inetflow.0 {
    maximum-prefixes 1000 threshold 50;
}
```

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

Limiting the Number of Prefixes Received on a BGP Peering Session

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.

```
set protocols bgp group x1 neighbor 10.12.99.2 family inet flow prefix-limit maximum 1000
set protocols bgp group x1 neighbor 10.12.99.2 family inet flow prefix-limit teardown 50
set protocols bgp group x1 neighbor 10.12.99.2 family inet flow prefix-limit drop-excess 50
set protocols bgp group x1 neighbor 10.12.99.2 family inet flow prefix-limit hide-excess 50
```

NOTE: You can include either the `teardown <percentage>`, `drop-excess <percentage>`, or `hide-excess<percentage>` statement option one at a time.

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 Junos OS CLI User Guide.

Configuring a prefix limit for a specific neighbor provides more predictable control over which peer can advertise how many flow routes.

To limit the number of prefixes:

1. Set a limit of 1000 BGP routes from neighbor 10.12.99.2.

```
[edit protocols bgp group x1]
user@host# set neighbor 10.12.99.2 family inet flow prefix-limit maximum 1000
```

2. Configure the neighbor session or prefixes to perform either `teardown <percentage>`, `drop-excess <percentage>`, or `hide-excess<percentage>` statement option when the session or prefixes reaches its limit.

```
[edit routing-options rib inetflow.0]
user@host# set neighbor 10.12.99.2 family inet flow prefix-limit teardown 50
set neighbor 10.12.99.2 family inet flow prefix-limit drop-excess 50
set neighbor 10.12.99.2 family inet flow prefix-limit hide-excess 50
```

If you specify the `teardown <percentage>` statement and specify a percentage, messages are logged when the number of prefixes reaches that percentage. After the session is brought down, the session reestablishes in a short time unless you include the `idle-timeout` statement.

If you specify the `drop-excess <percentage>` statement and specify a percentage, the excess routes are dropped when the number of prefixes exceeds that percentage

If you specify the `hide-excess <percentage>` statement and specify a percentage, the excess routes are hidden when the number of prefixes exceeds that percentage.

Results

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

```
[edit]
user@host# show protocols
bgp {
    group x1 {
        neighbor 10.12.99.2 {
                flow {
                    prefix-limit {
                        maximum 1000;
                        teardown 50;
                        drop-excess <percentage>;
                        hide-excess <percentage>;
                    }
                }
            }
        }
    }
}
```

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

Verification

IN THIS SECTION

Verifying the NLRI | **1138**

- Verifying Routes | **1139**
- Verifying Flow Validation | **1141**
- Verifying Firewall Filters | **1142**
- Verifying System Logging When Exceeding the Number of Allowed Flow Routes | **1143**
- Verifying System Logging When Exceeding the Number of Prefixes Received on a BGP Peering Session | **1144**

Confirm that the configuration is working properly.

Verifying the NLRI

Purpose

Look at the NLRI enabled for the neighbor.

Action

From operational mode, run the `show bgp neighbor 10.12.99.5` command. Look for `inet-flow` in the output.

```
user@host> show bgp neighbor 10.12.99.5
Peer: 10.12.99.5+3792 AS 65000 Local: 10.12.99.6+179 AS 65002
Type: External State: Established Flags: <Sync>
Last State: OpenConfirm Last Event: RecvKeepAlive
Last Error: None
Export: [ direct ]
Options: <Preference HoldTime AddressFamily PeerAS Refresh>
Address families configured: inet-unicast inet-multicast inet-flow
Holdtime: 90 Preference: 170
Number of flaps: 1
Error: 'Cease' Sent: 0 Recv: 1
Peer ID: 10.255.71.161 Local ID: 10.255.124.107 Active Holdtime: 90
Keepalive Interval: 30 Peer index: 0
Local Interface: e1-3/0/0.0
NLRI advertised by peer: inet-unicast inet-multicast inet-flow
NLRI for this session: inet-unicast inet-multicast inet-flow
Peer supports Refresh capability (2)
Table inet.0 Bit: 10000
```

```
RIB State: BGP restart is complete
Send state: in sync
Active prefixes: 2
Received prefixes: 2
Suppressed due to damping: 0
Advertised prefixes: 3
Table inet.2 Bit: 20000
RIB State: BGP restart is complete
Send state: in sync
Active prefixes: 0
Received prefixes: 0
Suppressed due to damping: 0
Advertised prefixes: 0
Table inetflow.0 Bit: 30000
RIB State: BGP restart is complete
Send state: in sync
Active prefixes: 0
Received prefixes: 0
Suppressed due to damping: 0
Advertised prefixes: 0
Last traffic (seconds): Received 29 Sent 15 Checked 15
Input messages: Total 5549 Updates 2618 Refreshes 0 Octets 416486
Output messages: Total 2943 Updates 1 Refreshes 0 Octets 55995
Output Queue[0]: 0
Output Queue[1]: 0
Output Queue[2]: 0
```

Verifying Routes

Purpose

Look at the flow routes. The sample output shows a flow route learned from BGP and a statically configured flow route.

For locally configured flow routes (configured at the `[edit routing-options flow]` hierarchy level), the routes are installed by the flow protocol. Therefore, you can display the flow routes by specifying the table, as in `show route table inetflow.0` or `show route table` *`instance-name`*`.inetflow.0`, where *`instance-name`* is the routing instance name. Or, you can display all locally configured flow routes across multiple routing instances by running the `show route protocol flow` command.

If a flow route is not locally configured, but received from the router's BGP peer, this flow route is installed in the routing table by BGP. You can display the flow routes by specifying the table or by running `show route protocol bgp`, which displays all BGP routes (flow and non-flow).

Action

From operational mode, run the `show route table inetflow.0` command.

```
user@host> show route table inetflow.0
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100.100.100.100,*,proto=1,icmp-type=8/term:1
                   *[BGP/170] 00:00:18, localpref 100, from 100.0.12.2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
200.200.200.200,*,proto=6,port=80/term:2
                   *[BGP/170] 00:00:18, localpref 100, from 100.0.12.2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
```

```
user@host> show route table inetflow.0 extensive
inetflow.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
7.7.7.7,8.8.8.8/term:1 (1 entry, 1 announced)
TSI:
KRT in dfwd;
Action(s): accept,count
        *Flow   Preference: 5
                Next hop type: Fictitious
                Address: 0x8d383a4
                Next-hop reference count: 3
                State: <Active>
                Local AS: 65000
                Age: 9:50
                Task: RT Flow
                Announcement bits (1): 0-Flow
                AS path: I
```

```
user@host> show route hidden
inetflow.0: 2 destinations, 2 routes (0 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

100.100.100.100,*,proto=1,icmp-type=8/term:N/A
                    [BGP ] 00:00:17, localpref 100, from 100.0.12.2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
200.200.200.200,*,proto=6,port=80/term:N/A
                    [BGP ] 00:00:17, localpref 100, from 100.0.12.2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
```

Meaning

A flow route represents a term of a firewall filter. When you configure a flow route, you specify the match conditions and the actions. In the match attributes, you can match a source address, a destination address, and other qualifiers such as the port and the protocol. For a single flow route that contains multiple match conditions, all the match conditions are encapsulated in the prefix field of the route. When you issue the `show route` command on a flow route, the prefix field of the route is displayed with all of the match conditions. `10.12.44.1,*` means that the matching condition is `match destination 10.12.44.1/32`. If the prefix in the output were `*,10.12.44.1`, this would mean that the match condition was `match source 10.12.44.1/32`. If the matching conditions contain both a source and a destination, the asterisk is replaced with the address.

The term-order numbers indicate the sequence of the terms (flow routes) being evaluated in the firewall filter. The `show route extensive` command displays the actions for each term (route).

Verifying Flow Validation

Purpose

Display flow route information.

Action

From operational mode, run the `show route flow validation detail` command.

```
user@host> show route flow validation detail
inet.0:
0.0.0.0/0
                Internal node: best match, inconsistent
10.0.0.0/8
                Internal node: no match, inconsistent
10.12.42.0/24
                Internal node: no match, consistent, next-as: 65003
                Active unicast route
                    Dependent flow destinations: 1
                    Origin: 10.255.124.106, Neighbor AS: 65003
10.12.42.1/32
                Flow destination (1 entries, 1 match origin)
                    Unicast best match: 10.12.42.0/24
                    Flags: Consistent
10.131.0.0/16
                Internal node: no match, consistent, next-as: 65001
                Active unicast route
                    Dependent flow destinations: 5000
                    Origin: 10.12.99.2, Neighbor AS: 65001
10.131.0.0/19
                Internal node: best match
10.131.0.0/20
                Internal node: best match
10.131.0.0/21
```

Verifying Firewall Filters

Purpose

Display the firewall filters that are installed in the kernel.

Action

From operational mode, run the `show firewall` command.

```
user@host> show firewall
Filter: __default_bpdu_filter__
Filter: __flowspec_default_inet__
Counters:
Name                                                Bytes              Packets
10.12.42.1,*                                            0                    0
196.1.28/23,*                                           0                    0
196.1.30/24,*                                           0                    0
196.1.31/24,*                                           0                    0
196.1.32/24,*                                           0                    0
196.1.56/21,*                                           0                    0
196.1.68/24,*                                           0                    0
196.1.69/24,*                                           0                    0
196.1.70/24,*                                           0                    0
196.1.75/24,*                                           0                    0
196.1.76/24,*                                           0                    0
```

Verifying System Logging When Exceeding the Number of Allowed Flow Routes

Purpose

If you configure a limit on the number of flow routes installed, as described in "Limiting the Number of Flow Routes Installed in a Routing Table" on page 1134, view the system log message when the threshold is reached.

Action

From operational mode, run the `show log <message>` command.

```
user@host> show log message
Jul 12 08:19:01  host rpd[2748]: RPD_RT_MAXROUTES_WARN: Number of routes (1000) in
table inetflow.0 exceeded warning threshold (50 percent of configured maximum 1000)
```

Verifying System Logging When Exceeding the Number of Prefixes Received on a BGP Peering Session

Purpose

If you configure a limit on the number of flow routes installed, as described in "Limiting the Number of Prefixes Received on a BGP Peering Session" on page 1136, view the system log message when the threshold is reached.

Action

From operational mode, run the **show log message** command.

If you specify the `teradown <percentage>` statement option:

```
user@host> show log message
Jul 12 08:44:47  host rpd[2748]: 10.12.99.2 (External AS 65001): Shutting down peer due to
exceeding configured maximum  prefix-limit(1000) for inet-flow nlri: 1001
```

If you specify the `drop-excess <percentage>` statement option:

```
user@host> show log message
Jul 27 15:26:57  R1_re rpd[32443]: BGP_DROP_PREFIX_LIMIT_EXCEEDED: 1.1.1.2 (Internal
AS 1): Exceeded drop-excess maximum prefix-limit(4) for inet-unicast nlri: 5 (instance master)
```

If you specify the `hide-excess <percentage>` statement option:

```
user@host> show log message
Jul 27 15:26:57  R1_re rpd[32443]: BGP_HIDE_PREFIX_LIMIT_EXCEEDED: 1.1.1.2 (Internal
AS 1): Exceeded hide-excess maximum prefix-limit(4) for inet-unicast nlri: 5 (instance master)
```

SEE ALSO

Understanding BGP Flow Routes for Traffic Filtering | 1115

Example: Configuring BGP to Carry IPv6 Flow Specification Routes

IN THIS SECTION

- Requirements | 1145
- Overview | 1145
- Configuration | 1146
- Verification | 1152

This example shows how to configure IPv6 flow specification for traffic filtering. BGP flow specification can be used to automate inter-domain and intra-domain coordination of traffic filtering rules in order to mitigate denial-of-service attacks.

Requirements

This example uses the following hardware and software components:

- Two MX Series routers
- Junos OS Release 16.1 or later

Before you enable BGP to carry IPv6 flow specification routes:

1. Configure IP addresses on the device interfaces.
2. Configure BGP.
3. Configure a routing policy that exports routes (such as static routes, direct routes, or IGP routes) from the routing table into BGP.

Overview

IN THIS SECTION

- Topology | 1146

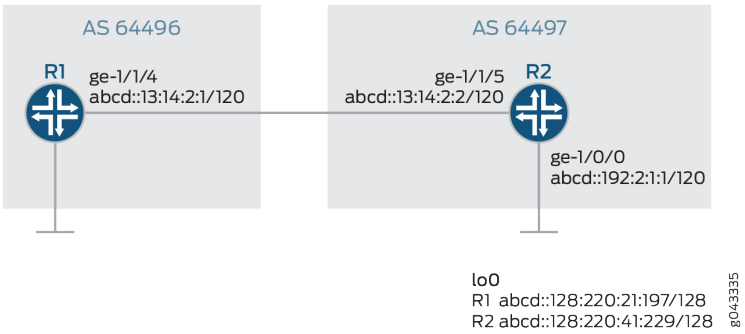
Flow specification provides protection against denial-of-service attacks and restricts bad traffic that consumes the bandwidth and stops it near the source. In earlier Junos OS releases, flow specification

rules were propagated for IPv4 over BGP as network layer reachability information. Beginning with Junos OS Release 16.1, the flow specification feature is supported on the IPv6 family and allows propagation of traffic flow specification rules for IPv6 and IPv6 VPN.

Topology

Figure 75 on page 1146 shows the sample topology. Router R1 and Router R2 belong to different autonomous systems. IPv6 flow specification is configured on Router R2. All incoming traffic is filtered based on the flow specification conditions, and the traffic is treated differently depending on the specified action. In this example, all traffic heading to abcd::11:11:11:10/128 that matches the flow specification conditions is discarded; whereas, traffic destined to abcd::11:11:11:30/128 and matching the flow specification conditions is accepted.

Figure 75: Configuring BGP to Carry IPv6 Flow Routes



Configuration

IN THIS SECTION

- CLI Quick Configuration | **1146**
- Configuring Router R2 | **1148**
- Results | **1150**

CLI Quick Configuration

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

Router R1

```
set interfaces ge-1/1/4 unit 0 family inet6 address abcd::13:14:2:1/120
set interfaces lo0 unit 0 family inet6 address abcd::128:220:21:197/128
set routing-options router-id 128.220.21.197
set routing-options autonomous-system 64496
set protocols bgp group ebgp type external
set protocols bgp group ebgp family inet6 unicast
set protocols bgp group ebgp family inet6 flow
set protocols bgp group ebgp peer-as 64497
set protocols bgp group ebgp neighbor abcd::13:14:2:2
```

Router R2

```
set interfaces ge-1/0/0 unit 0 family inet6 address abcd::192:2:1:1/120
set interfaces ge-1/1/5 unit 0 family inet6 address abcd::13:14:2:2/120
set interfaces lo0 unit 0 family inet6 address abcd::128:220:41:229/128
set routing-options rib inet6.0 static route abcd::11:11:11:0/120 next-hop abcd::192:2:1:2
set routing-options rib inet6.0 flow route route-1 match destination abcd::11:11:11:10/128
set routing-options rib inet6.0 flow route route-1 match protocol tcp
set routing-options rib inet6.0 flow route route-1 match destination-port http
set routing-options rib inet6.0 flow route route-1 match source-port 65535
set routing-options rib inet6.0 flow route route-1 then discard
set routing-options rib inet6.0 flow route route-2 match destination abcd::11:11:11:30/128
set routing-options rib inet6.0 flow route route-2 match icmp6-type echo-request
set routing-options rib inet6.0 flow route route-2 match packet-length 100
set routing-options rib inet6.0 flow route route-2 match dscp 10
set routing-options rib inet6.0 flow route route-2 then accept
set routing-options router-id 128.220.41.229
set routing-options autonomous-system 64497
set protocols bgp group ebgp type external
set protocols bgp group ebgp family inet6 unicast
set protocols bgp group ebgp family inet6 flow
set protocols bgp group ebgp export redis
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp neighbor abcd::13:14:2:1
set policy-options policy-statement redis from protocol static
set policy-options policy-statement redis then accept
```

Configuring Router R2

Step-by-Step Procedure

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

To configure Router R2:

NOTE: Repeat this procedure for Router R1 after modifying the appropriate interface names, addresses, and other parameters.

1. Configure the interfaces with IPv6 addresses.

```
[edit interfaces]
user@R2# set ge-1/0/0 unit 0 family inet6 address abcd::192:2:1:1/120
user@R2# set ge-1/1/5 unit 0 family inet6 address abcd::13:14:2:2/120
```

2. Configure the IPv6 loopback address.

```
[edit interfaces]
user@R2# set lo0 unit 0 family inet6 address abcd::128:220:41:229/128
```

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

```
[edit routing-options]
user@R2# set router-id 128.220.41.229
user@R2# set autonomous-system 64497
```

4. Configure an EBGP peering session between Router R1 and Router R2.

```
[edit protocols]
user@R2# set bgp group ebgp type external
user@R2# set bgp group ebgp family inet6 unicast
user@R2# set bgp group ebgp family inet6 flow
user@R2# set bgp group ebgp export redis
```

```
user@R2# set bgp group ebgp peer-as 64496
user@R2# set bgp group ebgp neighbor abcd::13:14:2:1
```

5. Configure a static route and a next hop. Thus a route is added to the routing table to verify the feature in this example.

```
[edit routing-options]
user@R2# set rib inet6.0 static route abcd::11:11:11:0/120 next-hop abcd::192:2:1:2
```

6. Specify flow specification conditions.

```
[edit routing-options]
user@R2# set rib inet6.0 flow route route-1 match destination abcd::11:11:11:10/128
user@R2# set rib inet6.0 flow route route-1 match protocol tcp
user@R2# set rib inet6.0 flow route route-1 match destination-port http
user@R2# set rib inet6.0 flow route route-1 match source-port 65535
```

7. Configure a **discard** action to discard packets that match the specified match conditions.

```
[edit routing-options]
user@R2# set rib inet6.0 flow route route-1 then discard
```

8. Specify flow specification conditions.

```
[edit routing-options]
user@R2# set rib inet6.0 flow route route-2 match destination abcd::11:11:11:30/128
user@R2# set rib inet6.0 flow route route-2 match icmp6-type echo-request
user@R2# set rib inet6.0 flow route route-2 match packet-length 100
user@R2# set rib inet6.0 flow route route-2 match dscp 10
```

9. Configure an **accept** action to accept packets that match the specified match conditions

```
[edit routing-options]
user@R2# set rib inet6.0 flow route route-2 then accept
```

10. Define a policy that allows BGP to accept static routes.

```
[edit policy-options]
user@R2# set policy-statement redis from protocol static
user@R2# set policy-statement redis then accept
```

Results

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

```
[edit]
user@R2# show interfaces
ge-1/0/0 {
    unit 0 {
        family inet6 {
            address abcd::192:2:1:1/120;
        }
    }
}
ge-1/1/5 {
    unit 0 {
        family inet6 {
            address abcd::13:14:2:2/120;
        }
    }
}
lo0 {
    unit 0 {
        family inet6 {
            address abcd::128:220:41:229/128;
        }
    }
}
```

```
[edit]
user@R2# show protocols
bgp {
```

```
    group ebgp {
        type external;
        family inet6 {
            unicast;
            flow;
        }
        export redis;
        peer-as 64496;
        neighbor abcd::13:14:2:1;
    }
}
```

```
[edit]
user@R2# show routing-options
rib inet6.0 {
    static {
        route abcd::11:11:11:0/120 next-hop abcd::192:2:1:2;
    }
    flow {
        route route-1 {
            match {
                destination abcd::11:11:11:10/128;
                protocol tcp;
                destination-port http;
                source-port 65535;
            }
            then discard;
        }
        route route-2 {
            match {
                destination abcd::11:11:11:30/128;
                icmp6-type echo-request;
                packet-length 100;
                dscp 10;
            }
            then accept;
        }
    }
}
```

```
router-id 128.220.41.229;
autonomous-system 64497;
```

```
[edit]
user@R2# show policy-options
policy-statement redis {
    from protocol static;
    then accept;
}
```

Verification

IN THIS SECTION

- Verifying the Presence of IPv6 Flow Specification Routes in the inet6flow Table | **1152**
- Verifying BGP Summary Information | **1155**
- Verifying Flow Validation | **1156**
- Verifying the Flow Specification of IPv6 Routes | **1157**

Confirm that the configuration is working properly.

Verifying the Presence of IPv6 Flow Specification Routes in the inet6flow Table

Purpose

Display the routes in the `inet6flow` table in Router R1 and R2, and verify that BGP has learned the flow routes.

Action

From operational mode, run the **show route table inet6flow.0 extensive** command on Router R1.

```
user@R1> show route table inet6flow.0 extensive
inet6flow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
abcd::11:11:11:10/128,*,proto=6,dstport=80,srcport=65535/term:1 (1 entry, 1 announced)
TSI:
```

```
KRT in dfwd;
Action(s): discard,count
        *BGP    Preference: 170/-101
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x9b24064
                Next-hop reference count: 2
                State:<Active Ext>
                Local AS:   64496 Peer AS:  64497
                Age: 20:55
                Validation State: unverified
                Task: BGP_64497.abcd::13:14:2:2
                Announcement bits (1): 0-Flow
                AS path: 64497 I
                Communities: traffic-rate:64497:0
                Accepted
                Validation state: Accept, Originator: abcd::13:14:2:2, Nbr AS: 64497
                Via: abcd::11:11:11:0/120, Active
                Localpref: 100
                Router ID: 128.220.41.229

abcd::11:11:11:30/128,*,icmp6-type=128,len=100,dscp=10/term:2 (1 entry, 1 announced)
TSI:
KRT in dfwd;
Action(s): accept,count
        *BGP    Preference: 170/-101
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x9b24064
                Next-hop reference count: 2
                State:  <Active Ext>
                Local AS:   64496 Peer AS:  64497
                Age: 12:51
                Validation State: unverified
                Task: BGP_64497.abcd::13:14:2:2
                Announcement bits (1): 0-Flow
                AS path: 64497 I
                Accepted
                Validation state: Accept, Originator: abcd::13:14:2:2, Nbr AS: 64497
                Via: abcd::11:11:11:0/120, Active
                Localpref: 100
                Router ID: 128.220.41.229
```

From operational mode, run the **show route table inet6flow.0 extensive** command on Router R2.

```
user@R2> show route table inet6flow.0 extensive
inet6flow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
abcd::11:11:11:10/128,*,proto=6,dstport=80,srcport=65535/term:1 (1 entry, 1 announced)
TSI:
KRT in dfwd;
Action(s): discard,count
Page 0 idx 0, (group pe-v6 type External) Type 1 val 0xaec8850 (adv_entry)
   Advertised metrics:
     Nexthop: Self
     AS path: [64497]
     Communities: traffic-rate:64497:0
Path abcd::11:11:11:10/128,*,proto=6,dstport=80,srcport=65535 Vector len 4.  Val: 0
        *Flow   Preference: 5
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x9b24064
                Next-hop reference count: 3
                State:  <Active>
                Local AS:  64497
                Age: 14:21
                Validation State: unverified
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP_RT_Background
                AS path: I
                Communities: traffic-rate:64497:0


abcd::11:11:11:30/128,*,proto=17,port=65535/term:2 (1 entry, 1 announced)
TSI:
KRT in dfwd;
Action(s): accept,count
Page 0 idx 0, (group pe-v6 type External) Type 1 val 0xaec8930 (adv_entry)
 Advertised metrics:
  Nexthop: Self
  AS path: [64497]
  Communities:
Path abcd::11:11:11:30/128,*,proto=17,port=65535 Vector len 4.  Val: 0
*Flow   Preference: 5
    Next hop type: Fictitious, Next hop index: 0
    Address: 0x9b24064
    Next-hop reference count: 3
```

```
    State: <Active>
    Local AS:  64497
    Age: 14:21
    Validation State: unverified
    Task: RT Flow
    Announcement bits (2): 0-Flow 1-BGP_RT_Background
    AS path: I
```

Meaning

The presence of routes abcd::11:11:11:10/128 and abcd::11:11:11:30/128 in the `inet6flow` table confirms that BGP has learned the flow routes.

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From operational mode, run the **show bgp summary** command on Router R1 and R2.

```
user@R1> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0
                       1          1          0          0          0          0
inet6flow.0
                       2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
abcd::13:14:2:2        2000         58         58       0       2       19:48 Establ
  inet6.0: 1/1/1/0
  inet6flow.0: 2/2/2/0
user@R2> show bgp summary
 Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet6.0
                       0          0          0          0          0          0
```

```
inet6flow.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
abcd::13:14:2:1         64496         51         52       0       0       23:03 Establ
  inet6.0: 0/0/0/0
  inet6flow.0: 0/0/0/0
```

Meaning

Verify that the `inet6.0` table contains the BGP neighbor address and a peering session has been established with its BGP neighbor.

Verifying Flow Validation

Purpose

Display flow route information.

Action

From operational mode, run the **show route flow validation** command on Router R1.

```
user@R1> show route flow validation
inet6.0:
abcd::11:11:11:0/120
                Active unicast route
                    Dependent flow destinations: 2
                    Origin: abcd::13:14:2:2, Neighbor AS: 64497
abcd::11:11:11:10/128
                Flow destination (1 entries, 1 match origin, next-as)
                    Unicast best match: abcd::11:11:11:0/120
                    Flags: Consistent
abcd::11:11:11:30/128
                Flow destination (1 entries, 1 match origin, next-as)
                    Unicast best match: abcd::11:11:11:0/120
                    Flags: Consistent
```

Meaning

The output displays the flow routes in the `inet6.0` table.

Verifying the Flow Specification of IPv6 Routes

Purpose

Display the number of packets that are discarded and accepted based on the specified flow specification routes.

Action

From operational mode, run the **show firewall filter_flowspec_default_inet6_** command on Router R2.

```
user@R2> show firewall filter __flowspec_default_inet6__
Filter: __flowspec_default_inet6__
Counters:
Name
                    Bytes              Packets
abcd::11:11:11:10/128,*,proto=6,dstport=80,srcport=65535           0                    0
abcd::11:11:11:30/128,*,proto=17,port=65535                           6395472
88826
```

Meaning

The output indicates that packets destined to abcd::11:11:11:10/128 are discarded and 88826 packets have been accepted for the route abcd::11:11:11:11:30/128.

SEE ALSO

flow

Configuring BGP Flow Specification Action Redirect to IP to Filter DDoS Traffic

Starting in Junos OS Release 18.4R1, BGP flow specification as described in BGP Flow-Spec Internet draft draft-ietf-idr-flowspec-redirect-ip-02.txt, *Redirect to IP Action* is supported. Redirect to IP action uses extended BGP community to provide traffic filtering options for DDoS mitigation in service provider networks. Legacy flow specification redirect to IP uses the BGP nexthop attribute. Junos OS advertises redirect to IP flow specification action using the extended community by default. This feature is required to support service chaining in virtual service control gateway (vSCG). Redirect to IP action allows to divert matching flow specification traffic to a globally reachable address that could be connected to a filtering device that can filter the DDoS traffic and send the clean traffic to the egress device.

Before you begin redirecting traffic to IP for BGP flow specification routes, do the following:

1. Configure the device interfaces.
2. Configure OSPF or any other IGP protocol.
3. Configure MPLS and LDP.
4. Configure BGP.

Configure the redirect to IP feature using the BGP extended community.

1. Configure redirect to IP action for static IPv4 flow specification routes as specified in the BGP Flow-Spec Internet draft draft-ietf-idr-flowspec-redirect-ip-02.txt, *Redirect to IP Action* .

   Junos OS advertises redirect to IP flow specification action using the extended community redirect to IP by default. The ingress device detects and sends the DDoS traffic to the specified IP address.

   ```
   [edit routing-options flow route then]
   user@host# set redirect ipv4-address
   ```

   For example, redirect the DDoS traffic to IPv4 address 10.1.1.1.

   ```
   [edit routing-options flow route then]
   user@host# set redirect 10.1.1.1.1
   ```

2. Configure redirect to IP action for static IPv6 flow specification routes.

```
[edit routing-options flow route then]
user@host# set redirect ipv6-address
```

For example, redirect the DDoS traffic to IPv6 address 1002:db8::

```
[edit routing-options flow route then]
user@host# set redirect 2001:db8::
```

3. Define a policy to filter traffic from a specific BGP community.

```
[edit policy-options]
user@host# policy-statement policy-name
user@host# from community community-ids
user@host# community community-ids members extended-community-type:administrator:assigned
number
```

For example, define a policy p1 to filter traffic from BGP community redirip.

```
[edit policy-options]
user@host# policy-statement p1
user@host# from community redirip
user@host# community redirip members redirect-to-ip :10.1.1.1:0
```

4. Define a policy to set, add, or delete a BGP community and specify the extended community.

```
[edit policy-options]
user@host# policy-statement policy-name
user@host# then community set community-ids
user@host# then community add community-ids
user@host# then community deletecommunity-ids
user@host# community community-ids members extended-community-type:administrator:assigned
number
```

For example, define a policy p1 to set, add, or delete a community reidirip and an extended community to redirect traffic to IP address 10.1.1.1.

```
[edit policy-options]
user@host# policy-statement p1
user@host# then community set redirip
user@host# then community add redirip
user@host# then community deleteredirip
user@host# community redirip members redirect-to-ip:10.1.1.1:0
```

5. Configure BGP to use VRF.inet.0 table to resolve VRF flow specification routes include statement at the hierarchy level.

```
[edit protocols bgp neighbor family flow]
user@host# set secondary-independent-resolution
```

Configure the legacy flow specification redirect to IP feature using the nexthop attribute.

NOTE: You cannot configure policies to redirect traffic to an IP address using BGP extended community and the legacy redirect to next hop IP address together.

1. Configure legacy flow specification redirect to IP specified in the internet draft draft-ietf-idr-flowspec-redirect-ip-00.txt , *BGP Flow-Spec Extended Community for Traffic Redirect to IP Next Hop* include at the hierarchy level.

```
[edit group bgp-group neighbor bgp neighbor family inet flow]
user@host# set legacy-redirect-ip-action
```

2. Define a policy to match the next hop attribute.

```
[edit policy options]
user@host#policy statement policy_name
user@host#from community community-name
user@host#from next-hop ip-address
```

For example, define a policy p1 to redirect traffic to next hop IP address 10.1.1.1.

```
[edit policy options]
user@host#policy statement p1
user@host#from community redirnh
user@host#from next-hop 10.1.1.1
```

3. Define a policy to set, add, or delete the BGP community using the legacy flow specification next hop attribute redirect to IP action.

```
[edit policy-options]
user@host# policy-statement policy_name
user@host# then community set community-name
user@host# then community add community-name
user@host# then community delete community-name
user@host# then next-hop next-hop-address
```

For example, define a policy p1 and set, add, or delete a BGP community redirnh to redirect the DDoS traffic to the the next hop IP address 10.1.1.1.

```
[edit policy-options policy-statement p1]
user@host# then community set redirnh
user@host# then community add redirnh
user@host# then community delete redirnh
user@host# then next-hop 10.1.1.1
```

SEE ALSO

Understanding BGP Flow Routes for Traffic Filtering | 1115

show route table

Forwarding Traffic Using BGP Flow Specification DSCP Action

IN THIS SECTION

- Benefits of BGP FlowSpec DSCP action to forward packets | 1162

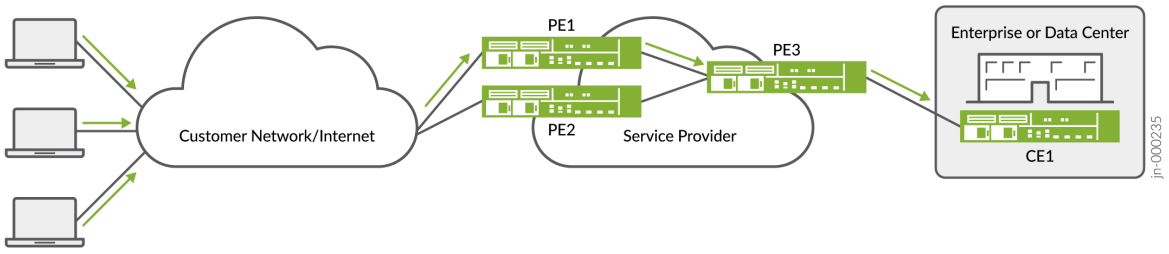
Configure BGP Flow Specification (FlowSpec) DSCP action to forward packets using the forwarding class and loss priority information across the network effectively.

Benefits of BGP FlowSpec DSCP action to forward packets

- Forwards traffic to the intended COS queues, where COS policies are applied to the traffic correctly.
- Influences local forwarding behavior (for example, selection of the tunnel) based on the provisioned DSCP value.
- Helps to manage traffic on your network effectively.

When a packet enters a router, the packet goes through the features (such as firewall, COS, etc.) applied at the ingress interface. When you configure BGP FlowSpec filter on the ingress interface, the filter is applied on the packets per routing instance based on the DSCP action. The DSCP action classifies and rewrites the packets, along with the DSCP code change through the BGP FlowSpec filter. Based on the forwarding class and loss priority information, the packets are placed to the correct forwarding queue. Packets travel through flow routes only if specific match conditions are met. The matching conditions can be source and destination IP address, source and destination port, DSCP, protocol number, etc. The forwarding class and loss priority information is updated through the reverse mapping table.

Here is a topology of a BGP session established between the service provider and the enterprise customer networks.



In this topology, a BGP session is configured between the service provider and the enterprise customer network for BGP FlowSpec. BGP FlowSpec filter is applied at both PE1 and PE2 routers. Packets entering these routers are rewritten based on the BGP FlowSpec filter and the DSCP action.

To enable the BGP FlowSpec filter on a device, you need to add the `dscp-mapping-classifier` configuration statement at the [`edit forwarding-options family (inet | inet6)`] hierarchy level:

```
forwarding-options {
    family inet {
        dscp-mapping-classifier ipv4-classifer;
    }
    family inet6 {
        dscp-mapping-classifier ipv6-classifer;
    }
}
```

The following sample class of service configuration maps DSCP code points to the forwarding class and loss priority:

```
class-of-service {
    classifiers {
        dscp dscp1 {
                forwarding-class best-effort {
                    loss-priority low code-points 000000;
                }
        }
      }
}
```

Configuring Policies for Flow Route Validation

When flow specification is used to distribute filters through the network to the peering points, the format of the flow specification filters is validated at the source. If you want signal flow routes over EBGP session to the peers, the flow specification filters must be validated at the edge routers.

Policies can be used to perform this additional validation of flow routes. A policy can filter the flow route match conditions to prevent admission of malformed, unsupported or undesired flow routes

injected from the source. It can also prevent flow routes from accidentally or maliciously blocking protocol sessions.

Policies can be used to

- Check if a particular flow specification match/action condition exists
- Prevent programming invalid or undesired match conditions.

Configuring Policies to Check if a flow specification match/action condition exists

You can configure policies to check whether a particular flow match/action condition is advertised from the source and take any necessary action. Each flow route has a match condition and an action condition. Configure the following statements specifying the match condition and the action to be taken if a match occurs.

```
set policy-options flowspec-attribute <> match condition <>

set policy-options flowspec-attribute < Flow Route action >
```

These policy configurations look for exact or partial matches in the flow route, and when found, the policy will match the flow route. If you configure multiple match conditions, then the matching occurs only when all the conditions are met.

You can use the 'invert' match option to exclude a certain match condition. For example; in the below configuration statements, the policy matches flow routes 'without tcp protocol and then matches 'with the dscp value 'x'. If both matches are found, then the policy accepts the flow routes.

```
set policy-options policy-statement flowspec-attribute fl1 invert-match protocol tcp

set policy-options policy-statement flowspec-attribute fl1 match dscp value <x>

set policy-options policy-statement pl1 term 1 from flowspec-attribute fl1
set policy-options policy-statement pl1 term 1 then accept
```

Configuring Policies to Check Flow Specification Match Attributes

You can configure policies to check whether the flow route matches the specific value of a flow match condition. This can prevent flow routes from accidentally or maliciously blocking protocol sessions.

Configure the following statement specifying the specific value of the attribute.

```
set policy-options flowspec-attribute < match attribute value >
```

Flow specification policy match attributes specific to IPv4 do not impact IPv6 flow routes and vice-versa.

Change History Table

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

Release	Description
20.3R1	Starting in cRPD Release 20.3R1, flow routes and policing rules propagated through BGP flow specification NLRI are downloaded to Linux kernel through Linux Netfilter framework on cRPD environments.
17.2R1	Starting in Junos OS Release 17.2R1, BGP can carry flow-specification network layer reachability information (NLRI) messages on PTX1000 routers that have third-generation FPCs installed.
17.1R1	Starting in Junos OS Release 17.1R1, BGP can carry flow-specification network layer reachability information (NLRI) messages on PTX Series routers that have third-generation FPCs (FPC3-PTX-U2 and FPC3-PTX-U3 on PTX5000 and FPC3-SFF-PTX-U0 and FPC3-SFF-PTX-U1 on PTX3000) installed.
16.1R4	Beginning with Junos OS Release 16.1R4, the rate-limit range is [0 through 1000000000000].
16.1	Beginning with Junos OS Release 16.1, IPv6 support is extended to BGP flow specification that allows propagation of traffic flow specification rules for IPv6 and VPN-IPv6 packets.
16.1	Starting with Junos OS Release 16.1R1, BGP flow specification supports traffic-marking `extended-community` filtering action.
16.1	Starting in Junos OS Release 16.1, you have the option to not apply the **flowspec** filter to traffic received on specific interfaces.
15.1	Beginning with Junos OS Release 15.1, changes are implemented to extend nonstop active routing (NSR) support for existing inet-flow and inetvpn-flow families and extend route validation for BGP flowspec per draft-ietf-idr-bgp-flowspec-oid-01.

8
CHAPTER

Configuring BGP CLNS

IN THIS CHAPTER

- BGP Connectionless Network Service (CLNS) | **1167**

BGP Connectionless Network Service (CLNS)

IN THIS SECTION

- Understanding BGP for CLNS VPNs | 1167
- Enabling BGP to Carry CLNS Routes | 1168
- Example: Configuring BGP for CLNS VPNs | 1174

Understanding BGP for CLNS VPNs

BGP extensions allow BGP to carry Connectionless Network Service (CLNS) virtual private network (VPN) network layer reachability information (NLRI) between provider edge (PE) routers. Each CLNS route is encapsulated into a CLNS VPN NLRI and propagated between remote sites in a VPN.

CLNS is a Layer 3 protocol similar to IP version 4 (IPv4). CLNS uses network service access points (NSAPs) to address end systems. This allows for a seamless autonomous system (AS) based on International Organization for Standardization (ISO) NSAPs.

A single routing domain consisting of ISO NSAP devices are considered to be CLNS islands. CLNS islands are connected together by VPNs.

You can configure BGP to exchange ISO CLNS routes between PE routers connecting various CLNS islands in a VPN using multiprotocol BGP extensions. These extensions are the ISO VPN NLRIs.

Each CLNS network island is treated as a separate VPN routing and forwarding instance (VRF) instance on the PE router.

You can configure CLNS on the global level, group level, and neighbor level.

SEE ALSO

CLNS Overview

Example: Configuring BGP for CLNS VPNs | 1174

Enabling BGP to Carry CLNS Routes

IN THIS SECTION

- Example: Enabling CLNS Between Two Routers | 1169
- Example: Configuring CLNS Within a VPN | 1171

Connectionless Network Service (CLNS) is a Layer 3 protocol similar to IP version 4 (IPv4). CLNS uses network service access points (NSAPs) to address end systems. This allows for a seamless autonomous system (AS) based on International Organization for Standardization (ISO) NSAPs.

Platform support for CLNS depends on the Junos OS release in your installation.

A single routing domain consisting of ISO NSAP devices are considered to be CLNS islands. CLNS islands are connected together by VPNs.

You can configure BGP to exchange ISO CLNS routes between provider edge (PE) routers connecting various CLNS islands in a virtual private network (VPN) using multiprotocol BGP extensions. These extensions are the ISO VPN NLRIs.

To enable multiprotocol BGP (MP-BGP) to carry CLNS VPN NLRIs, include the `iso-vpn` statement:

```
iso-vpn {
    unicast {
        prefix-limit number;
        rib-group group-name;
    }
}
```

To limit the number of prefixes from a peer, include the `prefix-limit` statement. To specify a routing table group, include the `rib-group` statement.

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

Each CLNS network island is treated as a separate VRF instance on the PE router.

You can configure CLNS on the global level, group level, and neighbor level.

For sample configurations, see the following sections:

Example: Enabling CLNS Between Two Routers

Configure CLNS between two routers through a route reflector:

```
On Router 1:
protocols {
    bgp {
        local-address 10.255.245.195;
        group pe-pe {
            type internal;
            neighbor 10.255.245.194 {
                family iso-vpn {
                    unicast;
                }
            }
        }
    }
}
routing-instances {
    aaaa {
        instance-type vrf;
        interface fe-0/0/0.0;
        interface so-1/1/0.0;
        interface lo0.1;
        route-distinguisher 10.255.245.194:1;
        vrf-target target:11111:1;
        protocols {
            isis {
                export dist-bgp;
                no-ipv4-routing;
                no-ipv6-routing;
                clns-routing;
                interface all;
            }
        }
    }
}
On Router 2:
protocols {
    bgp {
        group pe-pe {
            type internal;
```

```
            local-address 10.255.245.198;
            family route-target;
            neighbor 10.255.245.194 {
                family iso-vpn {
                    unicast;
                }
            }
        }
    }
}
routing-instances {
    aaaa {
        instance-type vrf;
        interface lo0.1;
        interface so-0/1/2.0;
        interface so-0/1/3.0;
        route-distinguisher 10.255.245.194:1;
        vrf-target target:11111:1;
        routing-options {
            rib aaaa.iso.0 {
                static {
                    iso-route 47.0005.80ff.f800.0000.bbbb.1022/104 next-hop
47.0005.80ff.f800.0000.aaaa.1000.1921.6800.4196.00;
                }
            }
        }
        protocols {
            isis {
                export dist-bgp;
                no-ipv4-routing;
                no-ipv6-routing;
                clns-routing;
                interface all;
            }
        }
    }
}
```

On Route Reflector:

```
protocols {
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.245.194;
```

```
            family route-target;
            neighbor 10.255.245.195 {
                cluster 0.0.0.1;
            }
            neighbor 10.255.245.198 {
                cluster 0.0.0.1;
            }
        }
    }
}
```

Example: Configuring CLNS Within a VPN

Configure CLNS on three PE routers within a VPN:

```
On PE Router 1:
protocols {
    mpls {
        interface all;
    }
    bgp {
        group asbr {
            type external;
            local-address 10.245.245.3;
            neighbor 10.245.245.1 {
                multihop;
                family iso-vpn {
                    unicast;
                }
                peer-as 200;
            }
        }
    }
}
routing-instances {
    aaaa {
        instance-type vrf;
        interface lo0.1;
        interface t1-3/0/0.0;
        interface fe-5/0/1.0;
        route-distinguisher 10.245.245.1:1;
```

```
        vrf-target target:11111:1;
        protocols {
            isis {
                export dist-bgp;
                no-ipv4-routing;
                no-ipv6-routing;
                clns-routing;
                interface all;
            }
        }
    }
}
```

On PE Router 2:

```
protocols {
    bgp {
        group asbr {
            type external;
            multihop;
            family iso-vpn {
                unicast;
            }
            neighbor 10.245.245.2 {
                peer-as 300;
            }
            neighbor 10.245.245.3 {
                peer-as 100;
            }
        }
    }
}
routing-instances {
    aaaa {
        instance-type vrf;
        interface lo0.1;
        route-distinguisher 10.245.245.1:1;
        vrf-target target:11111:1;
    }
}
```

On PE Router 3:

```
protocols {
    bgp {
        group asbr {
            type external;
```

```
            multihop;
            local-address 10.245.245.2;
            neighbor 10.245.245.1 {
                family iso-vpn {
                    unicast;
                }
                peer-as 200;
            }
        }
    }
}
routing-instances {
    aaaa {
        instance-type vrf;
        interface lo0.1;
        interface fe-0/0/1.0;
        interface t1-3/0/0.0;
        route-distinguisher 10.245.245.1:1;
        vrf-target target:11111:1;
        protocols {
            isis {
                export dist-bgp;
                no-ipv6-routing;
                clns-routing;
                interface all;
            }
        }
    }
}
```

SEE ALSO

- CLNS Overview

Example: Configuring BGP for CLNS VPNs

IN THIS SECTION

- Requirements | 1174
- Overview | 1174
- Configuration | 1174
- Verification | 1176

This example shows how to create a BGP group for CLNS VPNs, define the BGP peer neighbor address for the group, and define the family.

Requirements

Before you begin, configure the network interfaces. See the Interfaces User Guide for Security Devices.

Overview

In this example, you create the BGP group called pedge-pedge, define the BGP peer neighbor address for the group as 10.255.245.215, and define the BGP family.

Configuration

IN THIS SECTION

- Procedure | 1175

Procedure

CLI Quick Configuration

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

```
set protocols bgp group pedge-pedge neighbor 10.255.245.213
set protocols bgp family iso-vpn unicast
```

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 BGP for CLNS VPNs:

1. Configure the BGP group and define the BGP peer neighbor address.

```
[edit protocols bgp]
user@host# set group pedge-pedge neighbor 10.255.245.213
```

2. Define the family.

```
[edit protocols bgp]
user@host# set family iso-vpn unicast
```

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

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

Verification

IN THIS SECTION

- Verifying the Neighbor Status | **1176**

Verifying the Neighbor Status

Purpose

Display information about the BGP peer.

Action

From operational mode, run the `show bgp neighbor 10.255.245.213` command. Look for `iso-vpn-unicast` in the output.

```
user@host> show bgp neighbor 10.255.245.213
Peer: 10.255.245.213+179 AS 200 Local: 10.255.245.214+3770 AS 100
Type: External State: Established Flags: <ImportEval Sync>
Last State: OpenConfirm Last Event: RecvKeepAlive
Last Error: None
Options: <Multihop Preference LocalAddress HoldTime AddressFamily PeerAS
Rib-group Refresh>
Address families configured: iso-vpn-unicast
Local Address: 10.255.245.214 Holdtime: 90 Preference: 170
Number of flaps: 0
Peer ID: 10.255.245.213 Local ID: 10.255.245.214 Active Holdtime: 90
Keepalive Interval: 30 Peer index: 0
NLRI advertised by peer: iso-vpn-unicast
NLRI for this session: iso-vpn-unicast
Peer supports Refresh capability (2)
Table bgp.isovpn.0 Bit: 10000
RIB State: BGP restart is complete
RIB State: VPN restart is complete
Send state: in sync
Active prefixes: 3
Received prefixes: 3
Suppressed due to damping: 0
```

```
Advertised prefixes: 3
Table aaaa.iso.0
RIB State: BGP restart is complete
RIB State: VPN restart is complete
Send state: not advertising
Active prefixes: 3
Received prefixes: 3
Suppressed due to damping: 0
Last traffic (seconds): Received 6 Sent 5 Checked 5
Input messages: Total 1736 Updates 4 Refreshes 0 Octets 33385
Output messages: Total 1738 Updates 3 Refreshes 0 Octets 33305
Output Queue[0]: 0
Output Queue[1]: 0
```

SEE ALSO

- CLNS Configuration Overview
- Understanding BGP for CLNS VPNs | **1167**
- Verifying a CLNS VPN Configuration

9

CHAPTER

Using Route Reflectors and Confederations for BGP Networks

IN THIS CHAPTER

- BGP Route Reflectors | **1179**
- BGP Confederations for IBGP Scaling | **1221**

BGP Route Reflectors

IN THIS SECTION

- Understanding BGP Route Reflectors | **1179**
- Example: Configuring a Route Reflector | **1182**
- Understanding a Route Reflector That Belongs to Two Different Clusters | **1205**
- Example: Configuring a Route Reflector That Belongs to Two Different Clusters | **1206**
- Understanding BGP Optimal Route Reflection | **1213**
- Configuring BGP Optimal Route Reflection on a Route Reflector to Advertise the Best Path | **1215**
- BGP Route Server Overview | **1216**

Understanding BGP Route Reflectors

This topic discusses using route reflectors to simplify configuration and aid in scaling. A further way to reduce the workload on a route reflector that is not in the traffic-forwarding path is to use the `no-install` statement at the `[edit protocols bgp family family-name]` hierarchy level. Starting in Junos OS Release 15.1, the `no-install` statement eliminates interaction between the routing protocols daemon (rpd) and other components in the Junos system such as the kernel or the distributed firewall daemon (dfwd). This interaction is eliminated by prohibiting any routes in the associated rpd routing information bases (RIBs), also known as routing tables, from being published to those components.

NOTE: In releases previous to Junos OS Release 15.1, you can reduce the workload on a route reflector that is not in the traffic-forwarding path by using a forwarding-table export policy that rejects routes learned from BGP.

Because of the internal BGP (IBGP) full-mesh requirement, most networks use route reflectors to simplify configuration. The formula to compute the number of sessions required for a full mesh is v * (v - 1)/2, where v is the number of BGP-enabled devices. The full-mesh model does not scale well. Using a route reflector, you group routers into clusters, which are identified by numeric identifiers unique to the autonomous system (AS). Within the cluster, you must configure a BGP session from a single router (the route reflector) to each internal peer. With this configuration, the IBGP full-mesh requirement is met.

To use route reflection in an AS, you designate one or more routers as a route reflector—typically, one per point of presence (POP). Route reflectors have the special BGP ability to readvertise routes learned

from an internal peer to other internal peers. So rather than requiring all internal peers to be fully meshed with each other, route reflection requires only that the route reflector be fully meshed with all internal peers. The route reflector and all of its internal peers form a cluster, as shown in Figure 76 on page 1180.

NOTE: For some Juniper Networks devices, you must have an Advanced BGP Feature license installed on each device that uses a route reflector. For license details, see the Software Installation and Upgrade Guide.

Figure 76: Simple Route Reflector Topology (One Cluster)

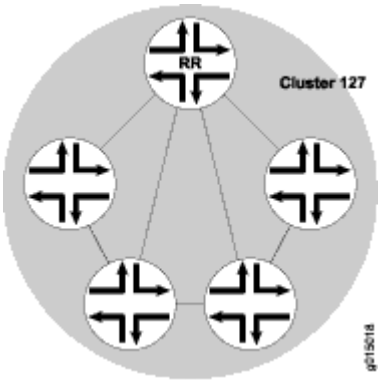


Figure 76 on page 1180 shows Router RR configured as the route reflector for Cluster 127. The other routers are designated internal peers within the cluster. BGP routes are advertised to Router RR by any of the internal peers. RR then readvertises those routes to all other peers within the cluster.

You can configure multiple clusters and link them by configuring a full mesh of route reflectors (see Figure 77 on page 1181).

Figure 77: Basic Route Reflection (Multiple Clusters)

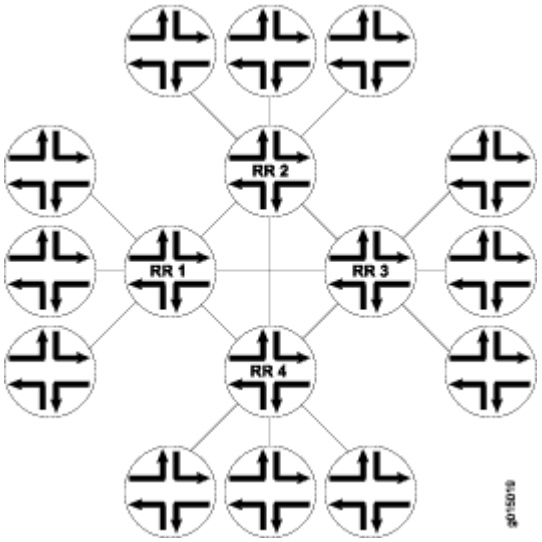


Figure 77 on page 1181 shows Route Reflectors RR 1, RR 2, RR 3, and RR 4 as fully meshed internal peers. When a router advertises a route to RR 1, RR 1 readvertises the route to the other route reflectors, which, in turn, readvertise the route to the remaining routers within the AS. Route reflection allows the route to be propagated throughout the AS without the scaling problems created by the full mesh requirement.

NOTE: A route reflector that supports multiple clusters does not accept a route with the same cluster ID from a non-client router. Therefore, you must configure a different cluster ID for a redundant RR to reflect the route to other clusters.

A client is an IBGP router that receives routes from the route reflector. A non-client is another IBGP neighbor. The route reflector adverises routes learned from non-clients only to its clients, not to non-clients. However, a route reflector advertises routes learned from clients to both clients and non-clients.

However, as clusters become large, a full mesh with a route reflector becomes difficult to scale, as does a full mesh between route reflectors. To help offset this problem, you can group clusters of routers together into clusters of clusters for hierarchical route reflection (see Figure 78 on page 1182).

Figure 78: Hierarchical Route Reflection (Clusters of Clusters)

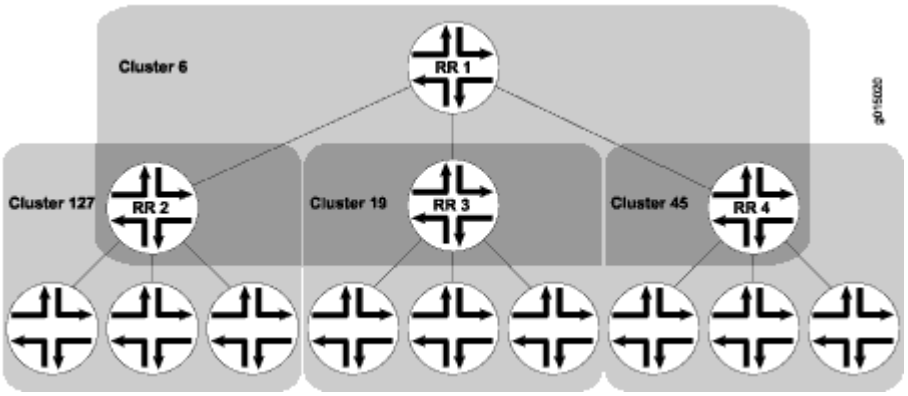


Figure 78 on page 1182 shows RR 2, RR 3, and RR 4 as the route reflectors for Clusters 127, 19, and 45, respectively. Rather than fully mesh those route reflectors, the network administrator has configured them as part of another cluster (Cluster 6) for which RR 1 is the route reflector. When a router advertises a route to RR 2, RR 2 readvertises the route to all the routers within its own cluster, and then readvertises the route to RR 1. RR 1 readvertises the route to the routers in its cluster, and those routers propagate the route down through their clusters.

SEE ALSO

- Understanding BGP | 2
- Installing Virtual Route Reflectors

Example: Configuring a Route Reflector

IN THIS SECTION

- Requirements | 1183
- Overview | 1183
- Configuration | 1184
- Verification | 1197

This example shows how to configure a route reflector.

Requirements

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

Overview

Generally, internal BGP (IBGP)-enabled devices need to be fully meshed, because IBGP does not readvertise updates to other IBGP-enabled devices. The full mesh is a logical mesh achieved through configuration of multiple `neighbor` statements on each IBGP-enabled device. The full mesh is not necessarily a physical full mesh. Maintaining a full mesh (logical or physical) does not scale well in large deployments.

Figure 79 on page 1184 shows an IBGP network with Device A acting as a route reflector. Device B and Device C are clients of the route reflector. Device D and Device E are outside the cluster, so they are nonclients of the route reflector.

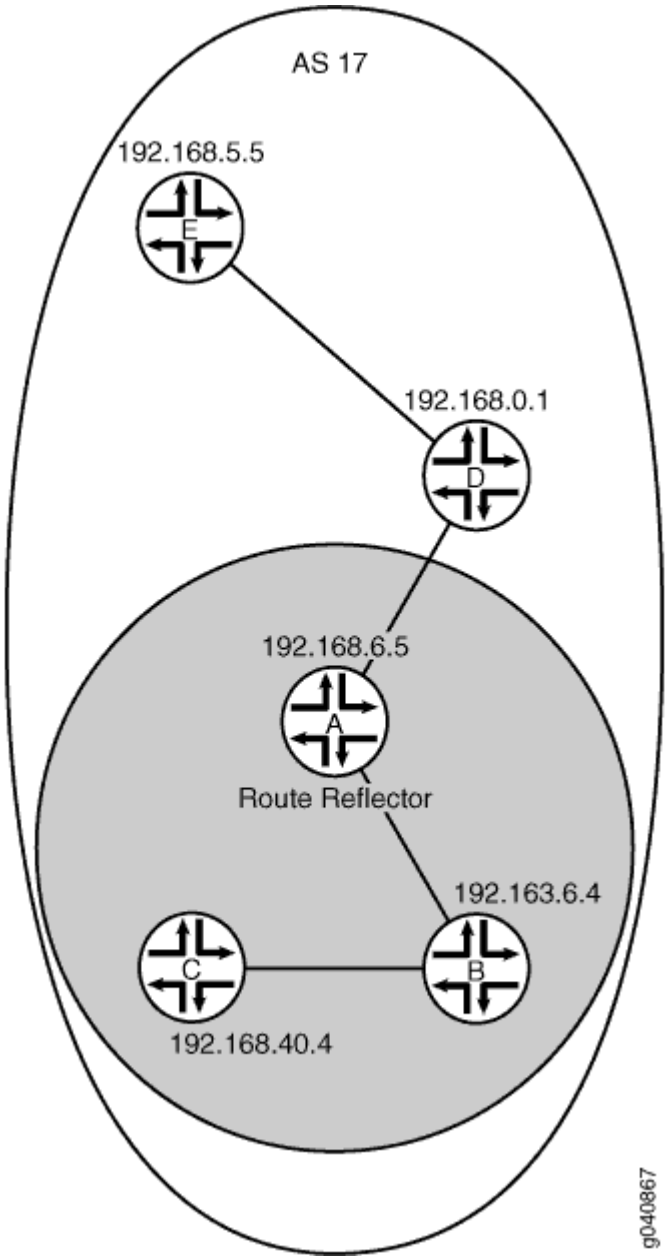
On Device A (the route reflector), you must form peer relationships with all of the IBGP-enabled devices by including the `neighbor` statement for the clients (Device B and Device C) and the nonclients (Device D and Device E). You must also include the `cluster` statement and a cluster identifier. The cluster identifier can be any 32-bit value. This example uses the loopback interface IP address of the route reflector.

On Device B and Device C, the route reflector clients, you only need one `neighbor` statement that forms a peer relationship with the route reflector, Device A.

On Device D and Device E, the nonclients, you need a `neighbor` statement for each nonclient device (D-to-E and E-to-D). You also need a `neighbor` statement for the route reflector (D-to-A and E-to-A). Device D and Device E do not need `neighbor` statements for the client devices (Device B and Device C).

TIP: Device D and Device E are considered to be nonclients because they have explicitly configured peer relationships with each other. To make them RRroute reflector clients, remove the `neighbor 192.168.5.5` statement from the configuration on Device D, and remove the `neighbor 192.168.0.1` statement from the configuration on Device E.

Figure 79: IBGP Network Using a Route Reflector



Configuration

IN THIS SECTION

- Procedure | **1185**
- Configuring the Route Reflector | **1187**

Configuring Client Peers | 1191
Configuring Nonclient Peers | 1194

Procedure

CLI Quick Configuration

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

Device A

```
set interfaces fe-0/0/0 unit 1 description to-B
set interfaces fe-0/0/0 unit 1 family inet address 10.10.10.1/30
set interfaces fe-0/0/1 unit 3 description to-D
set interfaces fe-0/0/1 unit 3 family inet address 10.10.10.9/30
set interfaces lo0 unit 1 family inet address 192.168.6.5/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 192.168.6.5
set protocols bgp group internal-peers export send-ospf
set protocols bgp group internal-peers cluster 192.168.6.5
set protocols bgp group internal-peers neighbor 192.163.6.4
set protocols bgp group internal-peers neighbor 192.168.40.4
set protocols bgp group internal-peers neighbor 192.168.0.1
set protocols bgp group internal-peers neighbor 192.168.5.5
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.1
set protocols ospf area 0.0.0.0 interface fe-0/0/1.3
set policy-options policy-statement send-ospf term 2 from protocol ospf
set policy-options policy-statement send-ospf term 2 then accept
set routing-options router-id 192.168.6.5
set routing-options autonomous-system 17
```

Device B

```
set interfaces fe-0/0/0 unit 2 description to-A
set interfaces fe-0/0/0 unit 2 family inet address 10.10.10.2/30
```

```
set interfaces fe-0/0/1 unit 5 description to-C
set interfaces fe-0/0/1 unit 5 family inet address 10.10.10.5/30
set interfaces lo0 unit 2 family inet address 192.163.6.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 192.163.6.4
set protocols bgp group internal-peers export send-ospf
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.2
set protocols ospf area 0.0.0.0 interface fe-0/0/1.5
set policy-options policy-statement send-ospf term 2 from protocol ospf
set policy-options policy-statement send-ospf term 2 then accept
set routing-options router-id 192.163.6.4
set routing-options autonomous-system 17
```

Device C

```
set interfaces fe-0/0/0 unit 6 description to-B
set interfaces fe-0/0/0 unit 6 family inet address 10.10.10.6/30
set interfaces lo0 unit 3 family inet address 192.168.40.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 192.168.40.4
set protocols bgp group internal-peers export send-ospf
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.6
set policy-options policy-statement send-ospf term 2 from protocol ospf
set policy-options policy-statement send-ospf term 2 then accept
set routing-options router-id 192.168.40.4
set routing-options autonomous-system 17
```

Device D

```
set interfaces fe-0/0/0 unit 4 description to-A
set interfaces fe-0/0/0 unit 4 family inet address 10.10.10.10/30
set interfaces fe-0/0/1 unit 7 description to-E
set interfaces fe-0/0/1 unit 7 family inet address 10.10.10.13/30
set interfaces lo0 unit 4 family inet address 192.168.0.1/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 192.168.0.1
set protocols bgp group internal-peers export send-ospf
```

```
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols bgp group internal-peers neighbor 192.168.5.5
set protocols ospf area 0.0.0.0 interface lo0.4 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.4
set protocols ospf area 0.0.0.0 interface fe-0/0/1.7
set policy-options policy-statement send-ospf term 2 from protocol ospf
set policy-options policy-statement send-ospf term 2 then accept
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 17
```

Device E

```
set interfaces fe-0/0/0 unit 8 description to-D
set interfaces fe-0/0/0 unit 8 family inet address 10.10.10.14/30
set interfaces lo0 unit 5 family inet address 192.168.5.5/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 192.168.5.5
set protocols bgp group internal-peers export send-ospf
set protocols bgp group internal-peers neighbor 192.168.0.1
set protocols bgp group internal-peers neighbor 192.168.6.5
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.8
set policy-options policy-statement send-ospf term 2 from protocol ospf
set policy-options policy-statement send-ospf term 2 then accept
set routing-options router-id 192.168.5.5
set routing-options autonomous-system 17
```

Step-by-Step Procedure

Configuring the Route Reflector

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 Junos OS CLI User Guide.

To configure IBGP in the network using Juniper Networks Device A as a route reflector:

1. Configure the interfaces.

```
[edit interfaces]
user@A# set fe-0/0/0 unit 1 description to-B
user@A# set fe-0/0/0 unit 1 family inet address 10.10.10.1/30
user@A# set fe-0/0/1 unit 3 description to-D
user@A# set fe-0/0/1 unit 3 family inet address 10.10.10.9/30
user@A# set lo0 unit 1 family inet address 192.168.6.5/32
```

2. Configure BGP, including the cluster identifier and neighbor relationships with all IBGP-enabled devices in the autonomous system (AS).

   Also apply the policy that redistributes OSPF routes into BGP.

```
[edit protocols bgp group internal-peers]
user@A# set type internal
user@A# set local-address 192.168.6.5
user@A# set export send-ospf
user@A# set cluster 192.168.6.5
user@A# set neighbor192.163.6.4
user@A# set neighbor 192.168.40.4
user@A# set neighbor 192.168.0.1
user@A# set neighbor 192.168.5.5
```

3. Configure static routing or an interior gateway protocol (IGP).

   This example uses OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@A# set interface lo0.1 passive
user@A# set interface fe-0/0/0.1
user@A# set interface fe-0/0/1.3
```

4. Configure the policy that redistributes OSPF routes into BGP.

```
[edit  policy-options policy-statement send-ospf term 2]
user@A# set from protocol ospf
user@A# set then accept
```

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

```
[edit routing-options]
user@A# set router-id 192.168.6.5
user@A# set autonomous-system 17
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@A# show interfaces
fe-0/0/0 {
    unit 1 {
        description to-B;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
fe-0/0/1 {
    unit 3 {
        description to-D;
        family inet {
            address 10.10.10.9/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.6.5/32;
        }
    }
}
```

```
user@A# show protocols
bgp {
```

```
    group internal-peers {
        type internal;
        local-address 192.168.6.5;
        export send-ospf;
        cluster 192.168.6.5;
        neighbor 192.163.6.4;
        neighbor 192.168.40.4;
        neighbor 192.168.0.1;
        neighbor 192.168.5.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface fe-0/0/0.1;
        interface fe-0/0/1.3;
    }
}
```

```
user@A# show policy-options
policy-statement send-ospf {
    term 2 {
        from protocol ospf;
        then accept;
    }
}
```

```
user@A# show routing-options
router-id 192.168.6.5;
autonomous-system 17;
```

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

NOTE: Repeat these steps for each nonclient BGP peer within the cluster that you are configuring, if the other nonclient devices are from Juniper Networks. Otherwise, consult the device's documentation for instructions.

Configuring Client Peers

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 Junos OS CLI User Guide.

To configure client peers:

1. Configure the interfaces.

```
[edit interfaces]
user@B# set fe-0/0/0 unit 2 description to-A
user@B# set fe-0/0/0 unit 2 family inet address 10.10.10.2/30
user@B# set fe-0/0/1 unit 5 description to-C
user@B# set fe-0/0/1 unit 5 family inet address 10.10.10.5/30
user@B# set lo0 unit 2 family inet address 192.163.6.4/32
```

2. Configure the BGP neighbor relationship with the route reflector.

   Also apply the policy that redistributes OSPF routes into BGP.

```
[edit protocols bgp group internal-peers]
user@B# set type internal
user@B# set local-address 192.163.6.4
user@B# set export send-ospf
user@B# set neighbor 192.168.6.5
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@B# set interface lo0.2 passive
user@B# set interface fe-0/0/0.2
user@B# set interface fe-0/0/1.5
```

4. Configure the policy that redistributes OSPF routes into BGP.

```
[edit  policy-options policy-statement send-ospf term 2]
user@B# set from protocol ospf
user@B# set then accept
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@B# set router-id 192.163.6.4
user@B# set autonomous-system 17
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@B# show interfaces
fe-0/0/0 {
    unit 2 {
        description to-A;
        family inet {
            address 10.10.10.2/30;
        }
    }
}
fe-0/0/1 {
    unit 5 {
        description to-C;
        family inet {
            address 10.10.10.5/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.163.6.4/32;
```

```
        }
    }
}
```

```
user@B# show protocols
bgp {
    group internal-peers {
        type internal;
        local-address 192.163.6.4;
        export send-ospf;
        neighbor 192.168.6.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface fe-0/0/0.2;
        interface fe-0/0/1.5;
    }
}
```

```
user@B# show policy-options
policy-statement send-ospf {
    term 2 {
        from protocol ospf;
        then accept;
    }
}
```

```
user@B# show routing-options
router-id 192.163.6.4;
autonomous-system 17;
```

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

NOTE: Repeat these steps for each client BGP peer within the cluster that you are configuring if the other client devices are from Juniper Networks. Otherwise, consult the device's documentation for instructions.

Configuring Nonclient Peers

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 Junos OS CLI User Guide.

To configure nonclient peers:

1. Configure the interfaces.

```
[edit interfaces]
user@D# set fe-0/0/0 unit 4 description to-A
user@D# set fe-0/0/0 unit 4 family inet address 10.10.10.10/30
user@D# set fe-0/0/1 unit 7 description to-E
user@D# set fe-0/0/1 unit 7 family inet address 10.10.10.13/30
user@D# set lo0 unit 4 family inet address 192.168.0.1/32
```

2. Configure the BGP neighbor relationships with the RRroute reflector and with the other nonclient peers.

   Also apply the policy that redistributes OSPF routes into BGP.

```
[edit protocols bgp group internal-peers]
user@D# set type internal
user@D# set local-address 192.168.0.1
user@D# set export send-ospf
user@D# set neighbor 192.168.6.5
user@D# set neighbor 192.168.5.5
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@D# set interface lo0.4 passive
```

```
user@D# set interface fe-0/0/0.4
user@D# set interface fe-0/0/1.7
```

4. Configure the policy that redistributes OSPF routes into BGP.

```
[edit  policy-options policy-statement send-ospf term 2]
user@D# set from protocol ospf
user@D# set then accept
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@D# set router-id 192.168.0.1
user@D# set autonomous-system 17
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@D# show interfaces
fe-0/0/0 {
    unit 4 {
        description to-A;
        family inet {
            address 10.10.10.10/30;
        }
    }
}
fe-0/0/1 {
    unit 7 {
        description to-E;
        family inet {
            address 10.10.10.13/30;
        }
    }
}
lo0 {
```

```
    unit 4 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@D# show protocols
bgp {
    group internal-peers {
        type internal;
        local-address 192.168.0.1;
        export send-ospf;
        neighbor 192.168.6.5;
        neighbor 192.168.5.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.4 {
            passive;
        }
        interface fe-0/0/0.4;
        interface fe-0/0/1.7;
    }
}
```

```
user@D# show policy-options
policy-statement send-ospf {
    term 2 {
        from protocol ospf;
        then accept;
    }
}
```

```
user@D# show routing-options
router-id 192.168.0.1;
autonomous-system 17;
```

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

> **NOTE**: Repeat these steps for each nonclient BGP peer within the cluster that you are configuring if the other nonclient devices are from Juniper Networks. Otherwise, consult the device's documentation for instructions.

Verification

IN THIS SECTION

- Verifying BGP Neighbors | **1197**
- Verifying BGP Groups | **1201**
- Verifying BGP Summary Information | **1202**
- Verifying Routing Table Information | **1202**

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is established for each neighbor address.

Action

From operational mode, enter the `show bgp neighbor` command.

```
user@A> show bgp neighbor
Peer: 192.163.6.4+179 AS 17    Local: 192.168.6.5+62857 AS 17
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-ospf ]
  Options: <Preference LocalAddress Cluster Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
```

```
  Peer ID: 192.163.6.4     Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            6
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 5    Sent 3    Checked 19
  Input messages:  Total 2961   Updates 7       Refreshes 0     Octets 56480
  Output messages: Total 2945   Updates 6       Refreshes 0     Octets 56235
  Output Queue[0]: 0

Peer: 192.168.0.1+179 AS 17    Local: 192.168.6.5+60068 AS 17
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-ospf ]
  Options: <Preference LocalAddress Cluster Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.0.1     Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 3
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
```

```
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            6
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 18   Sent 20   Checked 12
  Input messages:  Total 15     Updates 5       Refreshes 0     Octets 447
  Output messages: Total 554    Updates 4       Refreshes 0     Octets 32307
  Output Queue[0]: 0

Peer: 192.168.5.5+57458 AS 17  Local: 192.168.6.5+179 AS 17
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-ospf ]
  Options: <Preference LocalAddress Cluster Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.5.5     Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 2
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
```

```
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            7
    Accepted prefixes:            7
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 17   Sent 3    Checked 9
  Input messages:  Total 2967   Updates 7       Refreshes 0     Octets 56629
  Output messages: Total 2943   Updates 6       Refreshes 0     Octets 56197
  Output Queue[0]: 0

Peer: 192.168.40.4+53990 AS 17 Local: 192.168.6.5+179 AS 17
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ send-ospf ]
  Options: <Preference LocalAddress Cluster Refresh>
  Local Address: 192.168.6.5 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.40.4    Local ID: 192.168.6.5       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 1
  BFD: disabled, down
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Restart time configured on the peer: 120
  Stale routes from peer are kept for: 300
  Restart time requested by this peer: 120
  NLRI that peer supports restart for: inet-unicast
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 17)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
```

```
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            7
    Accepted prefixes:            7
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 5    Sent 23   Checked 52
  Input messages:  Total 2960   Updates 7       Refreshes 0     Octets 56496
  Output messages: Total 2943   Updates 6       Refreshes 0     Octets 56197
  Output Queue[0]: 0
```

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From operational mode, enter the `show bgp group` command.

```
user@A> show bgp group
Group Type: Internal    AS: 17                     Local AS: 17
  Name: internal-peers  Index: 0                   Flags: <>
  Export: [ send-ospf ]
  Options: <Cluster>
  Holdtime: 0
  Total peers: 4        Established: 4
  192.163.6.4+179
  192.168.40.4+53990
  192.168.0.1+179
  192.168.5.5+57458
  inet.0: 0/26/16/0

Groups: 1  Peers: 4    External: 0    Internal: 4    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                26          0          0          0          0          0
```

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From operational mode, enter the `show bgp summary` command.

```
user@A> show bgp summary
Groups: 1 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                26          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.163.6.4              17       2981       2965       0       0    22:19:15
0/6/1/0              0/0/0/0
192.168.0.1              17         36        575       0       0       13:43
0/6/1/0              0/0/0/0
192.168.5.5              17       2988       2964       0       0    22:19:10
0/7/7/0              0/0/0/0
192.168.40.4             17       2980       2964       0       0    22:19:14
0/7/7/0              0/0/0/0
```

Verifying Routing Table Information

Purpose

Verify that the routing table contains the IBGP routes.

Action

From operational mode, enter the `show route` command.

```
user@A> show route
inet.0: 12 destinations, 38 routes (12 active, 0 holddown, 10 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.10.0/30      *[Direct/0] 22:22:03
```

```
                    > via fe-0/0/0.1
                    [BGP/170] 22:20:55, MED 2, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via fe-0/0/0.1
                    [BGP/170] 22:20:51, MED 3, localpref 100, from 192.168.5.5
                      AS path: I
                    > to 10.10.10.10 via fe-0/0/1.3
10.10.10.1/32       *[Local/0] 22:22:03
                      Local via fe-0/0/0.1
10.10.10.4/30       *[OSPF/10] 22:21:13, metric 2
                    > to 10.10.10.2 via fe-0/0/0.1
                    [BGP/170] 22:20:51, MED 4, localpref 100, from 192.168.5.5
                      AS path: I
                    > to 10.10.10.10 via fe-0/0/1.3
10.10.10.8/30       *[Direct/0] 22:22:03
                    > via fe-0/0/1.3
                    [BGP/170] 22:20:51, MED 2, localpref 100, from 192.168.5.5
                      AS path: I
                    > to 10.10.10.10 via fe-0/0/1.3
                    [BGP/170] 22:20:55, MED 3, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via fe-0/0/0.1
10.10.10.9/32       *[Local/0] 22:22:03
                      Local via fe-0/0/1.3
10.10.10.12/30      *[OSPF/10] 22:21:08, metric 2
                    > to 10.10.10.10 via fe-0/0/1.3
                    [BGP/170] 22:20:55, MED 4, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via fe-0/0/0.1
192.163.6.4/32      *[OSPF/10] 22:21:13, metric 1
                    > to 10.10.10.2 via fe-0/0/0.1
                    [BGP/170] 22:20:55, MED 1, localpref 100, from 192.168.40.4
                      AS path: I
                    > to 10.10.10.2 via fe-0/0/0.1
                    [BGP/170] 22:20:51, MED 3, localpref 100, from 192.168.5.5
                      AS path: I
                    > to 10.10.10.10 via fe-0/0/1.3
192.168.0.1/32      *[OSPF/10] 22:21:08, metric 1
                    > to 10.10.10.10 via fe-0/0/1.3
                    [BGP/170] 22:20:51, MED 1, localpref 100, from 192.168.5.5
                      AS path: I
                    > to 10.10.10.10 via fe-0/0/1.3
                    [BGP/170] 22:20:55, MED 3, localpref 100, from 192.168.40.4
```

```
                    AS path: I
                  > to 10.10.10.2 via fe-0/0/0.1
192.168.5.5/32     *[OSPF/10] 22:21:08, metric 2
                  > to 10.10.10.10 via fe-0/0/1.3
                   [BGP/170] 00:15:24, MED 1, localpref 100, from 192.168.0.1
                    AS path: I
                  > to 10.10.10.10 via fe-0/0/1.3
                   [BGP/170] 22:20:55, MED 4, localpref 100, from 192.168.40.4
                    AS path: I
                  > to 10.10.10.2 via fe-0/0/0.1
192.168.6.5/32     *[Direct/0] 22:22:04
                  > via lo0.1
                   [BGP/170] 22:20:51, MED 2, localpref 100, from 192.168.5.5
                    AS path: I
                  > to 10.10.10.10 via fe-0/0/1.3
                   [BGP/170] 22:20:55, MED 2, localpref 100, from 192.168.40.4
                    AS path: I
                  > to 10.10.10.2 via fe-0/0/0.1
192.168.40.4/32    *[OSPF/10] 22:21:13, metric 2
                  > to 10.10.10.2 via fe-0/0/0.1
                   [BGP/170] 22:20:55, MED 1, localpref 100, from 192.163.6.4
                    AS path: I
                  > to 10.10.10.2 via fe-0/0/0.1
                   [BGP/170] 22:20:51, MED 4, localpref 100, from 192.168.5.5
                    AS path: I
                  > to 10.10.10.10 via fe-0/0/1.3
224.0.0.5/32       *[OSPF/10] 22:22:07, metric 1
                    MultiRecv
```

SEE ALSO

- Routing Policies, Firewall Filters, and Traffic Policers User Guide
- Example: Configuring Internal BGP Peer Sessions | **63**
- Example: Configuring External BGP Point-to-Point Peer Sessions | **27**
- Understanding BGP Route Reflectors | **1179**
- BGP Configuration Overview | **24**

Understanding a Route Reflector That Belongs to Two Different Clusters

The purpose of route reflection is loop prevention when the internal BGP (IBGP) routing devices are not fully meshed. To accomplish this, RRs break one of the rules of normal BGP operation: They readvertise routes learned from an internal BGP peer to other internal BGP peers.

Normally, a single RR is a member of only one cluster. Consider, for example, that in a hierarchical RR design, a tier-two RR can be the client of a tier-1 RR, but they can not be clients of each other.

However, when two RRs are clients of each other and the routes are being reflected from one cluster to another, only one of the cluster IDs is included in the cluster list. This is because having one cluster ID in the cluster list is adequate for loop prevention in this case.

Table 10 on page 1205 summarizes the rules that route reflectors use when filling in a reflected route's cluster list with cluster IDs.

Table 10: Rules for Route Reflectors

Route Reflection Scenario	Configuration
When reflecting a route from one of the clients to a non-client router client -> RR -> non-client	The RR fills the cluster ID associated with that client in the cluster list of the reflected route.
When reflecting a route from a non-client router to a client router non-client -> RR -> client	The RR fills the cluster ID associated with that client in the cluster list of the reflected route.
When reflecting a route from a client router to another client router that is in a different cluster client1 -> RR -> client2 (different cluster)	The RR fills the cluster ID associated with client1 in the cluster list before reflecting the cluster ID to client2. The cluster ID associated with client 2 is not added.

Table 10: Rules for Route Reflectors *(Continued)*

Route Reflection Scenario	Configuration
When reflecting a route from a client router to a non-client router that is in a different autonomous system. For example, when you have configured a tier 2 RR and 2 BGP groups, one for the RR clients and the other for tier 1 RR, and you are reflecting a route from one autonomous system to another autonomous system. client -> RR -> non-client (different AS)	The RR does not fill the cluster list with the cluster-ID before reflecting the route to the non-client device because the cluster-ID is specific to one autonomous system.

SEE ALSO

Understanding BGP Route Reflectors | 1179

Example: Configuring a Route Reflector That Belongs to Two Different Clusters

IN THIS SECTION

- Requirements | 1207
- Overview | 1207
- Configuration | 1207
- Verification | 1211

This example shows how to configure a route reflector (RR) that belongs to two different clusters. This is not a common scenario, but it might be useful in some situations.

Requirements

Configure the device interfaces and an internal gateway protocol (IGP). For an example of an RR setup that includes the interface and IGP configuration, see "Example: Configuring a Route Reflector" on page 1182.

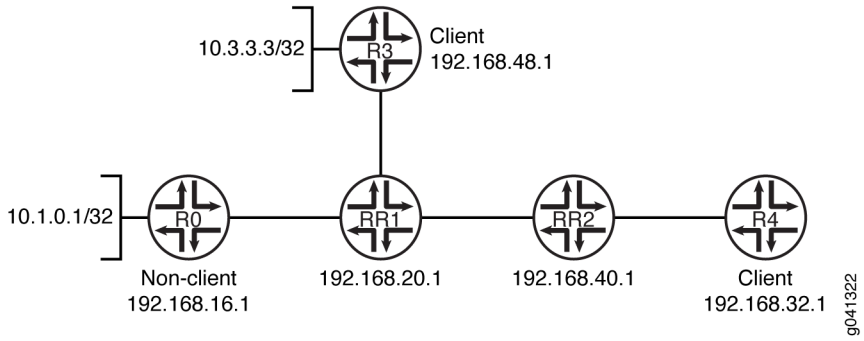
Overview

In this example, Device RR1 is a route reflector for both Device R3 and Device RR2. The route reflector RR1 has two different cluster IDs assigned, one is 10.13.1.3 for RR1-R3 and 10.12.1.2 for RR1-RR2.

Device RR2 is a route reflector for Device R4.

Consider figure Figure 80 on page 1207.

Figure 80: Route Reflector in Two Different Clusters



This example shows the BGP configuration on Device RR1 and Device RR2.

Configuration

IN THIS SECTION

- Procedure | **1208**
- Configuring Device RR1 | **1208**
- Configuring Device RR2 | **1210**

Procedure

CLI Quick Configuration

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

Device RR1

```
set protocols bgp group RR1_client type internal
set protocols bgp group RR1_client local-address 192.168.20.1
set protocols bgp group RR1_client cluster 10.13.1.3
set protocols bgp group RR1_client neighbor 192.168.48.1
set protocols bgp group Non_client type internal
set protocols bgp group Non_client local-address 192.168.20.1
set protocols bgp group Non_client neighbor 192.168.16.1
set protocols bgp group RR1_to_RR2 type internal
set protocols bgp group RR1_to_RR2 local-address 192.168.20.1
set protocols bgp group RR1_to_RR2 cluster 10.12.1.2
set protocols bgp group RR1_to_RR2 neighbor 192.168.40.1
```

Device RR2

```
set protocols bgp group RR2_client type internal
set protocols bgp group RR2_client local-address 192.168.40.1
set protocols bgp group RR2_client cluster 10.24.2.4
set protocols bgp group RR2_client neighbor 192.168.32.1
set protocols bgp group RR2_to_RR1 type internal
set protocols bgp group RR2_to_RR1 local-address 192.168.40.1
set protocols bgp group RR2_to_RR1 neighbor 192.168.20.1
```

Configuring Device RR1

Step-by-Step Procedure

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

To configure Device RR1:

1. Configure the peering relationship with Device R3.

```
[edit protocols bgp group RR1_client]
user@RR1# set type internal
user@RR1# set local-address 192.168.20.1
user@RR1# set cluster 10.13.1.3
user@RR1# set neighbor 192.168.48.1
```

2. Configure the peering relationship with Device R0.

```
[edit protocols bgp group Non_client]
user@RR1# set type internal
user@RR1# set local-address 192.168.20.1
user@RR1# set neighbor 192.168.16.1
```

3. Configure the peering relationship with Device RR2.

```
[edit protocols bgp group RR1_to_RR2]
user@RR1# set type internal
user@RR1# set local-address 192.168.20.1
user@RR1# set cluster 10.12.1.2
user@RR1# set neighbor 192.168.40.1
```

Results

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

```
user@RR1# show protocols
bgp {
    group RR1_client {
        type internal;
        local-address 192.168.20.1;
        cluster 10.13.1.3;
        neighbor 192.168.48.1;
    }
    group Non_client {
        type internal;
```

```
        local-address 192.168.20.1;
        neighbor 192.168.16.1;
    }
    group RR1_to_RR2 {
        type internal;
        local-address 192.168.20.1;
        cluster 10.12.1.2;
        neighbor 192.168.40.1;
    }
}
```

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

Configuring Device RR2

Step-by-Step Procedure

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

To configure Device RR2:

1. Configure the peering relationship with Device R4.

```
[edit protocols bgp group RR2_client]
user@RR2# set type internal
user@RR2# set local-address 192.168.40.1
user@RR2# set cluster 10.24.2.4
user@RR2# set neighbor 192.168.32.1
```

2. Configure the peering relationship with Device RR1.

```
[edit protocols bgp group RR2_to_RR1]
user@RR2# set type internal
user@RR2# set local-address 192.168.40.1
user@RR2# set neighbor 192.168.20.1
```

Results

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

```
user@RR2# show protocols
bgp {
    group RR2_client {
        type internal;
        local-address 192.168.40.1;
        cluster 10.24.2.4;
        neighbor 192.168.32.1;
    }
    group RR2_to_RR1 {
        type internal;
        local-address 192.168.40.1;
        neighbor 192.168.20.1;
    }
}
```

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

Verification

IN THIS SECTION

- Checking the Cluster ID Advertised for Route 10.3.3.3 | **1211**
- Checking the Cluster ID Advertised for Route 10.1.0.1 | **1212**

Confirm that the configuration is working properly.

Checking the Cluster ID Advertised for Route 10.3.3.3

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is established for each neighbor address.

Action

From operational mode, enter the `show route advertising-protocol bgp` *neighbor-address* command.

```
user@RR1> show route advertising-protocol bgp 192.168.40.1 active-path 10.3.3.3 extensive

inet.0: 61 destinations, 61 routes (60 active, 0 holddown, 1 hidden)
* 10.3.3.3/32 (1 entry, 1 announced)
 BGP group RR1_to_RR2 type Internal
     Nexthop: 192.168.48.1
     Localpref: 100
     AS path: [100] I
     Cluster ID: 10.13.1.3
     Originator ID: 192.168.48.1
```

Meaning

The 10.3.3.3/32 route originates from the Device RR1's client peer, Device R3. When this route is sent to RR1's client, Device RR2, the route has the 10.13.1.3 cluster ID attached, which is the cluster ID for RR1-R3.

Checking the Cluster ID Advertised for Route 10.1.0.1

Purpose

Check the route advertisement from Device RR1 to Device RR2.

Action

From operational mode, enter the `show route advertising-protocol bgp` *neighbor-address* command.

```
user@RR1> show route advertising-protocol bgp 192.168.40.1 active-path 10.1.0.1/32 extensive
inet.0: 61 destinations, 61 routes (60 active, 0 holddown, 1 hidden)
* 10.1.0.1/32 (1 entry, 1 announced)
 BGP group RR1_to_RR2 type Internal
     Nexthop: 192.168.16.1
     Localpref: 100
     AS path: [100] I
```

```
Cluster ID: 10.12.1.2
Originator ID: 192.168.16.1
```

Meaning

The 10.1.0.1/32 route originates from the Device RR1's non-client peer, Device R0. When this route is sent to RR1's client, Device RR2, the route has the 10.12.1.2 cluster ID attached, which is the cluster ID for RR1-RR2.

Device RR1 preserves the inbound cluster ID from Device RR2 when advertising to another client in a different cluster (Device R4).

SEE ALSO

Understanding a Route Reflector That Belongs to Two Different Clusters | 1205

Understanding BGP Optimal Route Reflection

You can configure BGP Optimal Route Reflection (BGP-ORR) with IS-IS and OSPF as the interior gateway protocol (IGP) on a route reflector to advertise the best path to the BGP-ORR client groups. This is done by using the shortest IGP metric from a client's perspective, instead of the route reflector's view.

Client groups sharing the same or similar IGP topology can be grouped as one BGP peer group. You can configure `optimal-route-reflection` to enable BGP-ORR in that BGP peer group. You can also configure one of the reflector nodes as the primary node (`igp-primary`) in a BGP peer group so that the IGP metric from that primary node is used to select the best path and advertise it to the clients in the same BGP peer group. Optionally, you can also select another reflector node as the backup node (`igp-backup`), which is used when the primary reflector node (`igp-primary`) goes down or is unreachable.

To enable BGP-ORR, include the `optimal-route-reflection` statement at the [`edit protocols bgp group` *group-name*] hierarchy level.

NOTE: BGP-ORR only works when IGP is used for BGP route resolution. BGP-ORR does not work when MPLSLDP/RSVP is used for route resolution.

NOTE: For BGP-ORR to work, the BGP prefix should be resolved through IGP. In normal Layer 3 VPN, or Layer 2 VPN, or VPLS, or MVPN, or EVPN scenarios, the prefixes are resolved over inet.3. In inet.3, the primary route for the protocol-next-hop of the prefix would be either RSVP or LDP (and not an IGP protocol like IS-IS or OSPF). For BGP-ORR to work, you need to configure the router to use inet.0 for the route-resolution of Layer 3 VPN, or Layer 2 VPN, or VPLS, or MVPN, or EVPN prefixes. You can do this through the following commands:

- For Layer 3 VPN prefix:

```
user@host# set routing-options resolution rib bgp.l3vpn.0 resolution-ribs inet.0
```

- For Layer 2 VPN, or VPLS prefix:

```
user@host# set routing-options resolution rib bgp.l2vpn.0 resolution-ribs inet.0
```

- For EVPN prefix:

```
user@host# set routing-options resolution rib bgp.evpn.0 resolution-ribs inet.0
```

- For MVPN prefix:

```
user@host# set routing-options resolution rib bgp.mvpn.0 resolution-ribs inet.0
```

Another method is to leak the IGP routes into inet.3 and make them as the primary route in inet.3.

Use the following CLI hierarchy to configure BGP-ORR:

```
[edit protocols bgp]
group group-name{
    optimal-route-reflection {
        igp-primary ipv4-address;
        igp-backup ipv4-address;
    }
}
```

SEE ALSO

Understanding BGP | 2

Configuring BGP Optimal Route Reflection on a Route Reflector to Advertise the Best Path

You can configure BGP Optimal Route Reflection (BGP-ORR) with IS-IS and OSPF as the interior gateway protocol (IGP) on a route reflector to advertise the best path to the BGP-ORR client groups. This is done by using the shortest IGP metric from a client's perspective, instead of the route reflector's view.

To enable BGP-ORR, include the `optimal-route-reflection` statement at the [`edit protocols bgp group` *`group-name`*] hierarchy level.

Client groups sharing the same or similar IGP topology can be grouped as one BGP peer group. You can configure `optimal-route-reflection` to enable BGP-ORR in that BGP peer group.

To configure BGP-ORR:

1. Configure optimal route reflection.

```
[edit protocols bgp group group-name]
user@host# set optimal-route-reflection
```

2. Configure one of the client nodes as the primary node (`igp-primary`) in a BGP peer group so that the IGP metric from that primary node is used to select the best path and advertise it to the clients in the same BGP peer group.

```
[edit protocols bgp group group-name optimal-route-reflection]
user@host# igp-primary ipv4-address;
```

3. (Optional) Configure another client node as the backup node (`igp-backup`), which is used when the primary node (`igp-primary`) goes down or is unreachable.

```
[edit protocols bgp group group-name optimal-route-reflection]
user@host# igp-backup ipv4-address;
```

Use the following CLI commands to monitor and troubleshoot the configuration for BGP-ORR:

- `show bgp group`—View the primary and backup configurations of BGP-ORR.

- `show isis bgp-orr`—View the IS-IS BGP-ORR metric (RIB).
- `show ospf bgp-orr`—View the OSPF BGP-ORR metric (RIB).
- `show ospf route`—View the entries in the OSPF routing table
- `show route`—View the active entries in the routing tables.
- `show route advertising protocol bgp` *peer*—Verify whether the routes are being advertised according to the BGP-ORR rules.

SEE ALSO

Understanding BGP | 2

Understanding BGP Optimal Route Reflection | 1213

BGP Route Server Overview

IN THIS SECTION

- BGP Attribute Transparency | 1217
- Next-Hop | 1218
- AS-Path | 1219
- Other Attributes | 1219
- BGP Route Server Client RIB | 1219
- Policy Requirements and Considerations | 1220

A BGP route server is the external BGP (EBGP) equivalent of an internal IBGP (IBGP) route reflector that simplifies the number of direct point-to-point EBGP sessions required in a network. EBGP route servers are transparent in terms of BGP attribute propagation so that a route received from a route server carries the set of BGP attributes (NEXT_HOP, AS_PATH, MULTI_EXIT_DISC, AIGP, and Communities) if the route is from a directly connected EBGP peer.

As with an IBGP route reflector, an EBGP route server is attached to a network outside of the direct forwarding path between the EBGP peers using the route server functionality. This connectivity can be through a direct physical link, or through Layer 2 networks such as VPLS LAN or EVPN, or through an IP

fabric of point-to-point EBGP links providing reachability of loopback addresses of peers (typical in data center networks).

The BGP protocol is enhanced to provide route-server capability with the following functionalities described in RFC 7947:

- Attribute transparency for NEXT_HOP, AS_PATH, MULTI_EXIT_DISC, AIGP, and Communities.
- Per-client policy control and multiple route-server RIBs for mitigation of path-hiding.

BGP programmability leverages the route-server functionality. The BGP JET **bgp_route_service.proto** API has been enhanced to support route server functionality as follows:

- Program the EBGP route server.
- Inject routes to the specific route server RIB for selectively advertising it to the client groups in client-specific RIBs.

The BGP JET **bgp_route_service.proto** API includes a peer-type object that identifies individual routes as either EBGP or IBGP (default).

Route server functionality is generally address-family independent, although certain specific BGP attribute support may be address-family-specific (for example, AIGP is only supported for labeled-unicast address-families). Route server functionality is supported for all EBGP address families.

BGP Attribute Transparency

EBGP attribute transparency [RFC7947] for the route server is supported by modifying the normal BGP route propagation such that neither transitive nor non-transitive attributes are stripped or modified while propagating routes.

Changes to normal EBGP behavior are controlled by the `route-server-client` CLI configuration. The `route-server-client` CLI configuration at the [`edit protocols bgp group` *`group-name`*] hierarchy level implements route server BGP attribute transparency behavior.

- The route server provides attribute transparency for both single-hop EBGP and multi-hop configurations. Therefore, the `route-server-client` configuration implicitly includes the functionality of `no-nexthop-change` for single-hop and multi-hop sessions. For multi-hop BGP sessions you must configure the `multihop` keyword.
- The `route-server-client` can be configured at the protocol, group, or neighbor levels of the [`edit protocol bgp`] hierarchy.
- The `route-server-client` configuration is applicable only when the group type is *external*. When the `route-server-client` is configured for *internal* groups, a configuration error is issued when attempting to commit.

- The `route-server-client` configuration has no parameters.
- Normal BGP privilege applies to the `route-server-client` configuration.

NOTE: Attributes are handled independently with respect to attribute transparency. Even if next-hops cannot be sent unmodified by the route-server, other attributes are sent transparently as applicable for those attributes.

The following is a sample `route-server-client` configuration:

```
[edit]
protocols {
    bgp {
        group R0 {
            type external;
                route-server-client;
                family inet {
                unicast;
                }
                peer-as 100;
            neighbor 10.0.0.1;
        }
    }
}
```

Next-Hop

The next-hop attribute must not be modified by imposing next-hop self or otherwise modifying the next-hop, unless explicitly configured through a policy. The route server must propagate BGP next-hops according to the third-party next-hop rules of RFC 4271.

Next-hop behavior is specified for the following route-server scenarios:

- In the case of LAN and WAN interconnect, when the route server is connected to client peers through a shared LAN and WAN subnet, the received third-party next-hops are advertised by the route server without modification as defined in RFC 4271.
- In the case of data center multihop interconnect, when the route server is connected to client peers through a multihop interconnect, EBGP multihop must be configured and the behavior of the `no-nexthop-change` CLI configuration is implicitly imposed by the `route-server-client` configuration. The received third-party next-hops are advertised by the route server without modification, as per the optional third-party behavior defined in RFC 4271.

- In other cases, such as point-to-point single-hop connections between the route server and client peers, normal next-hop advertisement procedures are used to prevent advertising next-hops that could be rejected by BGP peers (for example, most BGP implementations, by default, rejects next-hops addresses that are not covered by the subnet address on non-multipoint sessions.

AS-Path

AS-Path must not be modified by prepending the route server's local AS number. Configuring `route-server-client` CLI for a peer suppresses the prepending of the local AS number in the advertisements. In addition, the `show route advertising-protocol bgp` *peer* CLI command is changed for peers that are route server clients such that the local AS is not shown in the advertised metrics.

Other Attributes

- MULTI_EXIT_DISC attribute (optional, non-transitive) must be propagated as received.
- All community attributes, including no-advertise, no-export, and non-transitive extended communities, must be propagated as received.
- Accumulated IGP (AIGP) attribute (optional, non-transitive) must be propagated as received.

NOTE: Junos OS supports AIGP only for BGP-LU address families (IPv4 and IPv6).

BGP Route Server Client RIB

A route server client-specific RIB is a distinct view of a BGP Loc-RIB which can contain different routes for the same destination than other views. Route server clients, through their peer groups, may associate with one individual client-specific view or a shared common RIB.

In order to provide the ability to advertise different routes to different clients for the same destination, it is conceptually necessary to allow for multiple instances of the BGP path selection to occur for the same destination but in different client/group contexts.

To implement the high-level requirement of flexible policy control with per-client/group path selection, BGP route server takes the approach of using non-forwarding routing instances (NFIs) to multi-instance the BGP pipeline, including BGP path selection, Loc-RIB, and policy. The route server is configured to group route server clients within BGP groups configured within separate non-forwarding routing instances. This approach leverages the fact that BGP running within a routing instance does path selection and has a RIB that is separate from BGP running in other routing instances.

Policy Requirements and Considerations

To propagate routes between route server clients, routes are leaked between the RIBs of the routing instances based on configured policies.

Configuration of the route server for policy control includes the following considerations:

- Route server clients should be configured within the same primary instance or routing-instance to receive the same Loc-RIB view.
- Route server clients should be configured within their own routing-instance to receive totally unique Loc-RIB views.
- Route server clients should be configured in different BGP peer groups in the same routing-instance to have different export policy on the same Loc-RIB view.
- For the route server client-specific RIB views to receive all routes from other tables by default, a full-mesh of `instance-import` policies is configured with `instance-any`. When configuring `instance-import` with policy containing `instance-any`:
  - `instance-any` can be used in:
    - `policy-statement ... term ... from`
    - `policy-statement ... from`
    - `policy-statement ... term ... to`
    - `policy-statement ... to`
  - `instance-any` has no parameters.
  - Using `instance-any` in policies other than `instance-import` does not have any effect.
- Configuring many distinct routing-instances and peer-groups impacts scale and performance.
- The BGP `forwarding-context` CLI configuration at the [`edit protocols bgp group neighbor`] hierarchy level splits the routing instance for a BGP neighbor into a configuration instance and a forwarding instance. The BGP `forwarding-context` CLI configuration also supports non-forwarding instance with BGP peers configured as `route-server-client` where the specified instance is any instance capable of forwarding a primary or a routing-instance that is not of type no-forwarding.

SEE ALSO

Understanding BGP Optimal Route Reflection | 1213

Change History Table

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

Release	Description
15.1	Starting in Junos OS Release 15.1, the `no-install` statement eliminates interaction between the routing protocols daemon (rpd) and other components in the Junos system such as the kernel or the distributed firewall daemon (dfwd).
15.1	In releases previous to Junos OS Release 15.1, you can reduce the workload on a route reflector that is not in the traffic-forwarding path by using a forwarding-table export policy that rejects routes learned from BGP.

BGP Confederations for IBGP Scaling

IN THIS SECTION

- Understanding BGP Confederations | **1221**
- Example: Configuring BGP Confederations | **1223**

Understanding BGP Confederations

BGP confederations are another way to solve the scaling problems created by the BGP full mesh requirement. BGP confederations effectively break up a large autonomous system (AS) into subautonomous systems (sub-ASs). Each sub-AS must be uniquely identified within the confederation AS by a sub-AS number. Typically, sub-AS numbers are taken from the private AS numbers between 64,512 and 65,535.

Within a sub-AS, the same internal BGP (IBGP) full mesh requirement exists. Connections to other confederations are made with standard external BGP (EBGP), and peers outside the sub-AS are treated as external. To avoid routing loops, a sub-AS uses a confederation sequence, which operates like an AS path but uses only the privately assigned sub-AS numbers.

The confederation AS appears whole to other confederation ASs. The AS path received by other ASs shows only the globally assigned AS number. It does not include the confederation sequence or the

privately assigned sub-AS numbers. The sub-AS numbers are removed when the route is advertised out of the confederation AS. Figure 81 on page 1222 shows an AS divided into four confederations.

Figure 81: BGP Confederations

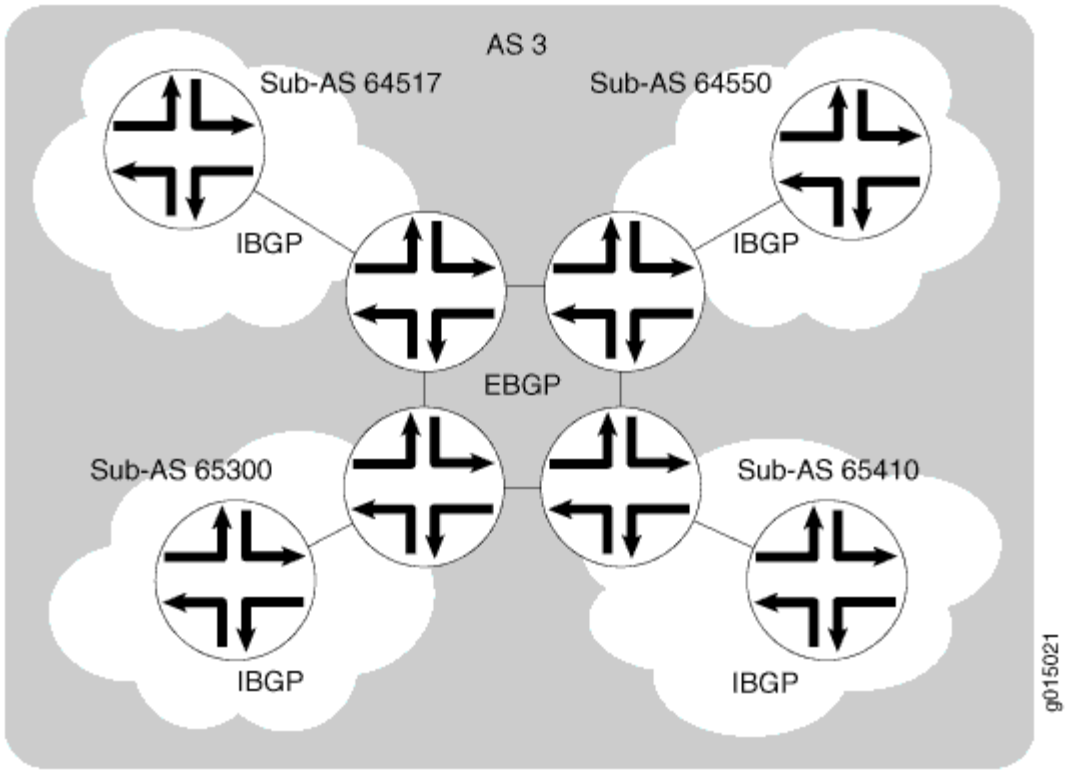


Figure 81 on page 1222 shows AS 3 divided into four sub-ASs, 64517, 64550, 65300, and 65410, which are linked through EBGP sessions. Because the confederations are connected by EBGP, they do not need to be fully meshed. EBGP routes are readvertised to other sub-ASs.

SEE ALSO

Understanding BGP | 2

Example: Configuring BGP Confederations

IN THIS SECTION

- Requirements | 1223
- Overview | 1223
- Configuration | 1224
- Verification | 1227

This example shows how to configure BGP confederations.

Requirements

- Configure network interfaces.
- Configure external peer sessions. See "Example: Configuring External BGP Point-to-Point Peer Sessions" on page 27.
- Configure interior gateway protocol (IGP) sessions between peers.
- Configure a routing policy to advertise the BGP routes.

Overview

IN THIS SECTION

- Topology | 1224

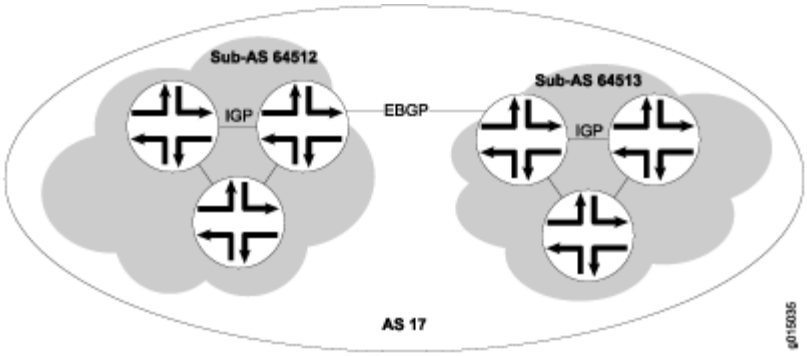
Within a BGP confederation, the links between the confederation member autonomous systems (ASs) must be external BGP (EBGP) links, not internal BGP (IBGP) links.

Similar to *route reflectors*, BGP confederations reduce the number of peer sessions and TCP sessions to maintain connections between IBGP routing devices. BGP confederation is one method used to solve the scaling problems created by the IBGP full mesh requirement. BGP confederations effectively break up a large AS into subautonomous systems. Each sub-AS must be uniquely identified within the confederation AS by a sub-AS number. Typically, sub-AS numbers are taken from the private AS numbers between 64512 and 65535. Within a sub-AS, the same IBGP full mesh requirement exists.

Connections to other confederations are made with standard EBGP, and peers outside the sub-AS are treated as external. To avoid routing loops, a sub-AS uses a confederation sequence, which operates like an AS path but uses only the privately assigned sub-AS numbers.

Figure 82 on page 1224 shows a sample network in which AS 17 has two separate confederations: sub-AS 64512 and sub-AS 64513, each of which has multiple routers. Within a sub-AS, an IGP is used to establish network connectivity with internal peers. Between sub-ASs, an EBGP peer session is established.

Figure 82: Typical Network Using BGP Confederations



Topology

Configuration

IN THIS SECTION

- Procedure | **1224**

Procedure

CLI Quick Configuration

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

All Devices in Sub-AS 64512

```
set routing-options autonomous-system 64512
set routing-options confederation 17 members 64512
set routing-options confederation 17 members 64513
set protocols bgp group sub-AS-64512 type internal
set protocols bgp group sub-AS-64512 local-address 192.168.5.1
set protocols bgp group sub-AS-64512 neighbor 192.168.8.1
set protocols bgp group sub-AS-64512 neighbor 192.168.15.1
```

Border Device in Sub-AS 64512

```
set protocols bgp group to-sub-AS-64513 type external
set protocols bgp group to-sub-AS-64513 peer-as 64513
set protocols bgp group to-sub-AS-64513 neighbor 192.168.5.2
```

All Devices in Sub-AS 64513

```
set routing-options autonomous-system 64513
set routing-options confederation 17 members 64512
set routing-options confederation 17 members 64513
set protocols bgp group sub-AS-64513 type internal
set protocols bgp group sub-AS-64513 local-address 192.168.5.2
set protocols bgp group sub-AS-64513 neighbor 192.168.9.1
set protocols bgp group sub-AS-64513 neighbor 192.168.16.1
```

Border Device in Sub-AS 64513

```
set protocols bgp group to-sub-AS-64512 type external
set protocols bgp group to-sub-AS-64512 peer-as 64512
set protocols bgp group to-sub-AS-64512 neighbor 192.168.5.1
```

Step-by-Step Procedure

This procedure shows the steps for the devices that are in sub-AS 64512.

The `autonomous-system` statement sets the sub-AS number of the device.

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 Junos OS CLI User Guide.

To configure BGP confederations:

1. Set the sub-AS number for the device.

```
[edit routing-options]
user@host# set autonomous-system 64512
```

2. In the confederation, include all sub-ASs in the main AS.

   The number 17 represents the main AS. The `members` statement lists all the sub-ASs in the main AS.

```
[edit routing-options confederation]
user@host# set 17 members 64512
user@host# set 17 members 64513
```

3. On the border device in sub-AS 64512, configure an EBGP connection to the border device in AS 64513.

```
[edit protocols bgp group to-sub-AS-64513]
user@host# set type external
user@host# set neighbor 192.168.5.2
user@host# set peer-as 64513
```

4. Configure an IBGP group for peering with the devices within sub-AS 64512.

```
[edit protocols bgp group sub-AS-64512]
user@host# set type internal
user@host# set local-address 192.168.5.1
user@host# neighbor 192.168.8.1
user@host# neighbor 192.168.15.1
```

Results

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

```
user@host# show routing-options
autonomous-system 64512;
confederation 17 members [ 64512 64513 ];
```

```
user@host# show protocols
bgp {
    group to-sub-AS-64513 { # On the border devices only
        type external;
        peer-as 64513;
        neighbor 192.168.5.2;
    }
    group sub-AS-64512 {
        type internal;
        local-address 192.168.5.1;
        neighbor 192.168.8.1;
        neighbor 192.168.15.1;
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.
Repeat these steps for sSub-AS 64513.

Verification

IN THIS SECTION

- Verifying BGP Neighbors | **1228**
- Verifying BGP Groups | **1229**
- Verifying BGP Summary Information | **1230**

Confirm that the configuration is working properly.

Verifying BGP Neighbors

Purpose

Verify that BGP is running on configured interfaces and that the BGP session is active for each neighbor address.

Action

From the CLI, enter the `show bgp neighbor` command.

Sample Output

command-name

```
user@host> show bgp neighbor
Peer: 10.255.245.12+179 AS 35  Local: 10.255.245.13+2884 AS 35
  Type: Internal    State: Established  (route reflector client)Flags: Sync
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: Preference LocalAddress HoldTime Cluster AddressFamily Rib-group Refresh
  Address families configured: inet-vpn-unicast inet-labeled-unicast
  Local Address: 10.255.245.13 Holdtime: 90 Preference: 170
  Flags for NLRI inet-vpn-unicast: AggregateLabel
  Flags for NLRI inet-labeled-unicast: AggregateLabel
  Number of flaps: 0
  Peer ID: 10.255.245.12    Local ID: 10.255.245.13    Active Holdtime: 90
  Keepalive Interval: 30
  NLRI advertised by peer: inet-vpn-unicast inet-labeled-unicast
  NLRI for this session: inet-vpn-unicast inet-labeled-unicast
  Peer supports Refresh capability (2)
Restart time configured on the peer: 300
  Stale routes from peer are kept for: 60
  Restart time requested by this peer: 300
  NLRI that peer supports restart for: inet-unicast inet6-unicast
  NLRI that restart is negotiated for: inet-unicast inet6-unicast
  NLRI of received end-of-rib markers: inet-unicast inet6-unicast
  NLRI of all end-of-rib markers sent: inet-unicast inet6-unicast
  Table inet.0 Bit: 10000
    RIB State: restart is complete
    Send state: in sync
```

```
    Active prefixes: 4
    Received prefixes: 6
    Suppressed due to damping: 0
  Table inet6.0 Bit: 20000
    RIB State: restart is complete
    Send state: in sync
    Active prefixes: 0
    Received prefixes: 2
    Suppressed due to damping: 0
  Last traffic (seconds): Received 3    Sent 3    Checked 3
  Input messages:  Total 9      Updates 6       Refreshes 0     Octets 403
  Output messages: Total 7      Updates 3       Refreshes 0     Octets 365
  Output Queue[0]: 0
  Output Queue[1]: 0
  Trace options: detail packets
  Trace file: /var/log/bgpgr size 131072 files 10
```

Meaning

The output shows a list of the BGP neighbors with detailed session information. Verify the following information:

- Each configured peering neighbor is listed.
- For `State`, each BGP session is `Established`.
- For `Type`, each peer is configured as the correct type (either internal or external).
- For `AS`, the AS number of the BGP neighbor is correct.

Verifying BGP Groups

Purpose

Verify that the BGP groups are configured correctly.

Action

From the CLI, enter the `show bgp group` command.

Sample Output

command-name

```
user@host> show bgp group
Group Type: Internal    AS: 10045       Local AS: 10045
  Name: pe-to-asbr2                                  Flags: Export Eval
  Export: [ match-all ]
  Total peers: 1        Established: 1
  10.0.0.4+179
  bgp.l3vpn.0: 1/1/0
  vpn-green.inet.0: 1/1/0

Groups: 1   Peers: 1    External: 0    Internal: 1    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0            1          1          0          0          0          0
```

Meaning

The output shows a list of the BGP groups with detailed group information. Verify the following information:

- Each configured group is listed.
- For `AS`, each group's remote AS is configured correctly.
- For `Local AS`, each group's local AS is configured correctly.
- For `Group Type`, each group has the correct type (either internal or external).
- For `Total peers`, the expected number of peers within the group is shown.
- For `Established`, the expected number of peers within the group have BGP sessions in the `Established` state.
- The IP addresses of all the peers within the group are present.

Verifying BGP Summary Information

Purpose

Verify that the BGP configuration is correct.

Action

From the CLI, enter the `show bgp summary` command.

Sample Output

command-name

```
user@host> show bgp summary
Groups: 1 Peers: 3 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 6          4          0          0          0          0
Peer               AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/Received/
Damped...
10.0.0.2        65002      88675      88652       0       2       42:38 2/4/0
0/0/0
10.0.0.3        65002      54528      54532       0       1     2w4d22h 0/0/0
0/0/0
10.0.0.4        65002      51597      51584       0       0     2w3d22h 2/2/0
0/0/0
```

Meaning

The output shows a summary of BGP session information. Verify the following information:

- For `Groups`, the total number of configured groups is shown.
- For `Peers`, the total number of BGP peers is shown.
- For `Down Peers`, the total number of unestablished peers is 0. If this value is not zero, one or more peering sessions are not yet established.
- Under `Peer`, the IP address for each configured peer is shown.
- Under `AS`, the peer AS for each configured peer is correct.
- Under `Up/Dwn State`, the BGP state reflects the number of paths received from the neighbor, the number of these paths that have been accepted, and the number of routes being damped (such as 0/0/0). If the field is `Active`, it indicates a problem in the establishment of the BGP session.

SEE ALSO

Routing Policies, Firewall Filters, and Traffic Policers User Guide

BGP Configuration Overview | 24

10

CHAPTER

Configuring BGP Security

IN THIS CHAPTER

- BGP Route Authentication | **1234**
- IP Security for BGP | **1245**
- TCP Access Restriction for BGP | **1251**
- BGP Origin Validation | **1274**

BGP Route Authentication

IN THIS SECTION

- Understanding Router Authentication for BGP | 1234
- TCP Authentication | 1235
- Example: Configuring Router Authentication for BGP | 1236

Understanding Router Authentication for BGP

The use of router and route authentication and route integrity greatly mitigates the risk of being attacked by a machine or router that has been configured to share incorrect routing information with another router. In this kind of attack, the attacked router can be tricked into creating a routing loop, or the attacked router's routing table can be greatly increased thus impacting performance, or routing information can be redirected to a place in the network for the attacker to analyze it. Bogus route advertisements can be sent out on a segment. These updates can be accepted into the routing tables of neighbor routers unless an authentication mechanism is in place to verify the source of the routes.

Router and route authentication enables routers to share information only if they can verify that they are talking to a trusted source, based on a password (key). In this method, a hashed key is sent along with the route being sent to another router. The receiving router compares the sent key to its own configured key. If they are the same, it accepts the route. By using a hashing algorithm, the key is not sent over the wire in plain text. Instead, a hash is calculated using the configured key. The routing update is used as the input text, along with the key, into the hashing function. This hash is sent along with the route update to the receiving router. The receiving router compares the received hash with a hash it generates on the route update using the preshared key configured on it. If the two hashes are the same, the route is assumed to be from a trusted source. The key is known only to the sending and receiving routers.

To further strengthen security, you can configure a series of authentication keys (a *keychain*). Each key has a unique start time within the keychain. Keychain authentication allows you to change the password information periodically without bringing down peering sessions. This keychain authentication method is referred to as *hitless* because the keys roll over from one to the next without resetting any peering sessions or interrupting the routing protocol.

The sending peer uses the following rules to identify the active authentication key:

- The start time is less than or equal to the current time (in other words, not in the future).

- The start time is greater than that of all other keys in the chain whose start time is less than the current time (in other words, closest to the current time).

The receiving peer determines the key with which it authenticates based on the incoming key identifier.

The sending peer identifies the current authentication key based on a configured start time and then generates a hash value using the current key. The sending peer then inserts a TCP-enhanced authentication option object into the BGP update message. The object contains an object ID (assigned by IANA), the object length, the current key, and a hash value.

The receiving peer examines the incoming TCP-enhanced authentication option, looks up the received authentication key, and determines whether the key is acceptable based on the start time, the system time, and the tolerance parameter. If the key is accepted, the receiving peer calculates a hash and authenticates the update message.

Initial application of a keychain to a TCP session causes the session to reset. However, once the keychain is applied, the addition or removal of a password from the keychain does not cause the TCP session to reset. Also, the TCP session does not reset when the keychain changes from one authentication algorithm to another.

SEE ALSO

> *Example: Configuring Hitless Authentication Key Rollover for IS-IS*
>
> *Example: Configuring MD5 Authentication for OSPFv2 Exchanges*

TCP Authentication

IN THIS SECTION

- TCP Authentication and Prefix Subnets | **1236**

Typically, you configure TCP authentication at the following hierarchy levels:

- `[edit protocols bgp]`
- `[edit protocols bgp group` *`group-name`*`]`
- `[edit protocols bgp group` *`group-name`* `neighbor` *`address`*`]`

TCP Authentication and Prefix Subnets

Junos devices support TCP authentication to BGP peers that are discovered through allowed prefix subnets configured in a BGP group.

To configure prefix-based authentication for TCP-AO or TCP MD5 for BGP sessions, you can configure the `allow (all | prefix-list)` statement at the following hierarchies:

- `[edit protocols bgp group group-name]`
- `[edit protocols bgp group group-name dynamic-neighbor dyn-name]`

For more information about TCP authentication, see *TCP*.

Example: Configuring Router Authentication for BGP

IN THIS SECTION

- Requirements | **1236**
- Overview | **1237**
- Configuration | **1238**
- Verification | **1241**

All BGP protocol exchanges can be authenticated to guarantee that only trusted routing devices participate in autonomous system (AS) routing updates. By default, authentication is disabled.

Requirements

Before you begin:

- Configure the router interfaces.
- Configure an interior gateway protocol (IGP).

Overview

IN THIS SECTION

- Topology Diagram | 1238

When you configure authentication, the algorithm creates an encoded checksum that is included in the transmitted packet. The receiving routing device uses an authentication key (password) to verify the packet's checksum.

This example includes the following statements for configuring and applying the keychain:

- `key`—A keychain can have multiple keys. Each key within a keychain must be identified by a unique integer value. The range of valid identifier values is from 0 through 63.

  The key can be up to 126 characters long. Characters can include any ASCII strings. If you include spaces, enclose all characters in quotation marks (“ ”).

- `tolerance`—(Optional) For each keychain, you can configure a clock-skew tolerance value in seconds. The clock-skew tolerance is applicable to the receiver accepting keys for BGP updates. The configurable range is 0 through 999,999,999 seconds. During the tolerance period, either the current or previous password is acceptable.

- `key-chain`—For each keychain, you must specify a name. This example defines one keychain: `bgp-auth`. You can have multiple keychains on a routing device. For example, you can have a keychain for BGP, a keychain for OSPF, and a keychain for LDP.

- `secret`—For each key in the keychain, you must set a secret password. This password can be entered in either encrypted or plain text format in the `secret` statement. It is always displayed in encrypted format.

- `start-time`—Each key must specify a start time in UTC format. Control gets passed from one key to the next. When a configured start time arrives (based on the routing device's clock), the key with that start time becomes active. Start times are specified in the local time zone for a routing device and must be unique within the keychain.

- `authentication-key-chain`—Enables you to apply a keychain at the global BGP level for all peers, for a group, or for a neighbor. This example applies the keychain to the peers defined in the external BGP (EBGP) group called `ext`.

- `authentication-algorithm`—For each keychain, you can specify a hashing algorithm. The algorithm can be AES-128, MD5, or SHA-1.

You associate a keychain and an authentication algorithm with a BGP neighboring session.

This example configures a keychain named `bgp-auth`. Key 0 will be sent and accepted starting at 2011-6-23.20:19:33 -0700, and will stop being sent and accepted when the next key in the keychain (key 1) becomes active. Key 1 becomes active one year later at 2012-6-23.20:19:33 -0700, and will not stop being sent and accepted unless another key is configured with a start time that is later than the start time of key 1. A clock-skew tolerance of 30 seconds applies to the receiver accepting the keys. During the tolerance period, either the current or previous key is acceptable. The keys are shared-secret passwords. This means that the neighbors receiving the authenticated routing updates must have the same authentication keychain configuration, including the same keys (passwords). So Router R0 and Router R1 must have the same authentication-key-chain configuration if they are configured as peers. This example shows the configuration on only one of the routing devices.

Topology Diagram

Figure 83 on page 1238 shows the topology used in this example.

Figure 83: Authentication for BGP

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1239**
- Procedure | **1239**

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.

```
set protocols bgp group ext type external
set protocols bgp group ext peer-as 65530
set protocols bgp group ext neighbor 172.16.2.1
set routing-options autonomous-system 65533
set protocols bgp group ext authentication-key-chain bgp-auth
set protocols bgp group ext authentication-algorithm md5
set security authentication-key-chains key-chain bgp-auth tolerance 30
set security authentication-key-chains key-chain bgp-auth key 0 secret this-is-the-secret-
password
set security authentication-key-chains key-chain bgp-auth key 0 start-time
2011-6-23.20:19:33-0700
set security authentication-key-chains key-chain bgp-auth key 1 secret this-is-another-secret-
password
set security authentication-key-chains key-chain bgp-auth key 1 start-time
2012-6-23.20:19:33-0700
```

Procedure

Step-by-Step Procedure

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

To configure Router R1 to accept route filters from Device CE1 and perform outbound route filtering using the received filters:

1. Configure the local autonomous system.

```
[edit routing-options]
user@R1# set autonomous-system 65533
```

2. Configure one or more BGP groups.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set peer-as 65530
user@R1# set neighbor 172.16.2.1
```

3. Configure authentication with multiple keys.

```
[edit security authentication-key-chains key-chain bgp-auth]
user@R1# set key 0 secret this-is-the-secret-password
user@R1# set key 0 start-time 2011-6-23.20:19:33-0700
user@R1# set key 1 secret this-is-another-secret-password
user@R1# set key 1 start-time 2012-6-23.20:19:33-0700
```

   The start time of each key must be unique within the keychain.

4. Apply the authentication keychain to BGP, and set the hashing algorithm.

```
[edit protocols bgp group ext]
user@R1# set authentication-key-chain bgp-auth
user@R1# set authentication-algorithm md5
```

5. (Optional) Apply a clock-skew tolerance value in seconds.

```
[edit security authentication-key-chains key-chain bgp-auth]
user@R1# set tolerance 30
```

Results

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

```
user@R1# show protocols
bgp {
    group ext {
        type external;
```

```
        peer-as 65530;
        neighbor 172.16.2.1;
        authentication-key-chain bgp-auth;
        authentication-algorithm md5;
    }
}
```

```
user@R1# show routing-options
autonomous-system 65533;
```

```
user@R1# show security
authentication-key-chains {
    key-chain bgp-auth {
        tolerance 30;
            key 0 {
            secret $ABC123$ABC123
            start-time “2011-6-23.20:19:33 -0700”;
        }
        key 1 {
            secret $ABC123$ABC123
            start-time “2012-6-23.20:19:33 -0700”;
        }
    }
}
```

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

Repeat the procedure for every BGP-enabled device in the network, using the appropriate interface names and addresses for each BGP-enabled device.

Verification

IN THIS SECTION

- Verifying Authentication for the Neighbor | **1242**
- Verifying That Authorization Messages Are Sent | **1243**
- Checking Authentication Errors | **1244**
- Verifying the Operation of the Keychain | **1244**

Confirm that the configuration is working properly.

Verifying Authentication for the Neighbor

Purpose

Make sure that the `AutheKeyChain` option appears in the output of the `show bgp neighbor` command.

Action

From operational mode, enter the `show bgp neighbor` command.

```
user@R1> show bgp neighbor
Peer: 172.16.2.1+179 AS 65530  Local: 172.16.2.2+1222 AS 65533
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ direct-lo0 ]
  Options: <Preference PeerAS Refresh>
  Options: <AutheKeyChain>
  Authentication key is configured
  Authentication key chain: jni
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 172.16.2.1       Local ID: 10.255.124.35    Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  Local Interface: fe-0/0/1.0
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              2
    Received prefixes:            2
    Suppressed due to damping:    0
    Advertised prefixes:          1
  Last traffic (seconds): Received 2    Sent 2    Checked 2
  Input messages:  Total 21     Updates 2       Refreshes 0     Octets 477
  Output messages: Total 22     Updates 1       Refreshes 0     Octets 471
  Output Queue[0]: 0
```

Verifying That Authorization Messages Are Sent

Purpose

Confirm that BGP has the enhanced authorization option.

Action

From operational mode, enter the `monitor traffic interface fe-0/0/1` command.

```
user@R1> monitor traffic interface fe-0/0/1
verbose output suppressed, use <detail> or <extensive> for full protocol decode
Listening on fe-0/0/1, capture size 96 bytes

13:08:00.618402  In arp who-has 172.16.2.66 tell 172.16.2.69
13:08:02.408249 Out IP 172.16.2.2.1122 > 172.16.2.1.646: P
1889289217:1889289235(18) ack 2215740969 win 58486 <nop,nop,timestamp 167557
1465469,nop,Enhanced Auth keyid 0 diglen 12 digest: fe3366001f45767165f17037>:
13:08:02.418396  In IP 172.16.2.1.646 > 172.16.2.2.1122: P 1:19(18) ack 18 win
57100 <nop,nop,timestamp 1466460 167557,nop,Enhanced Auth keyid 0 diglen 12
digest: a18c31eda1b14b2900921675>:
13:08:02.518146 Out IP 172.16.2.2.1122 > 172.16.2.1.646: . ack 19 win 58468
<nop,nop,timestamp 167568 1466460,nop,Enhanced Auth keyid 0 diglen 12 digest:
c3b6422eb6bd3fd9cf79742b>
13:08:28.199557 Out IP 172.16.2.2.nerv > 172.16.2.1.bgp: P
286842489:286842508(19) ack 931203976 win 57200 <nop,Enhanced Auth keyid 0
diglen 12 digest: fc0e42900a73736bcc07c1a4>: BGP, length: 19
13:08:28.209661  In IP 172.16.2.1.bgp > 172.16.2.2.nerv: P 1:20(19) ack 19 win
56835 <nop,Enhanced Auth keyid 0 diglen 12 digest: 0fc8578c489fabce63aeb2c3>:
BGP, length: 19
13:08:28.309525 Out IP 172.16.2.2.nerv > 172.16.2.1.bgp: . ack 20 win 57181
<nop,Enhanced Auth keyid 0 diglen 12 digest: ef03f282fb2ece0039491df8>
13:08:32.439708 Out IP 172.16.2.2.1122 > 172.16.2.1.646: P 54:72(18) ack 55 win
58432 <nop,nop,timestamp 170560 1468472,nop,Enhanced Auth keyid 0 diglen 12
digest: 76e0cf926f348b726c631944>:
13:08:32.449795  In IP 172.16.2.1.646 > 172.16.2.2.1122: P 55:73(18) ack 72 win
57046 <nop,nop,timestamp 1469463 170560,nop,Enhanced Auth keyid 0 diglen 12
digest: dae3eec390d18a114431f4d8>:
13:08:32.549726 Out IP 172.16.2.2.1122 > 172.16.2.1.646: . ack 73 win 58414
<nop,nop,timestamp 170571 1469463,nop,Enhanced Auth keyid 0 diglen 12 digest:
851df771aee2ea7a43a0c46c>
13:08:33.719880  In arp who-has 172.16.2.66 tell 172.16.2.69
```

```
^C
35 packets received by filter
0 packets dropped by kernel
```

Checking Authentication Errors

Purpose

Check the number of packets dropped by TCP because of authentication errors.

Action

From operational mode, enter the `show system statistics tcp | match auth` command.

```
user@R1> show system statistics tcp | match auth
        0 send packets dropped by TCP due to auth errors
        58 rcv packets dropped by TCP due to auth errors
```

Verifying the Operation of the Keychain

Purpose

Check the number of packets dropped by TCP because of authentication errors.

Action

From operational mode, enter the `show security keychain detail` command.

```
user@R1> show security keychain detail
keychain                Active-ID       Next-ID       Transition  Tolerance
                      Send  Receive   Send  Receive
 bgp-auth               3     3        1     1        1d 23:58    30
  Id 3, Algorithm hmac-md5, State send-receive, Option basic
  Start-time Wed Aug 11 16:28:00 2010, Mode send-receive
  Id 1, Algorithm hmac-md5, State inactive, Option basic
  Start-time Fri Aug 20 11:30:57 2010, Mode send-receive
```

SEE ALSO

Understanding External BGP Peering Sessions | 26

BGP Configuration Overview | 24

Change History Table

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

Release	Description
22.4R1	Starting in Junos OS Evolved Release 22.4R1, you can configure TCP-AO or TCP MD5 authentication with an IP subnet to include the entire range of addresses under that subnet.
22.4R1	Starting in Junos OS Evolved Release 22.4R1, TCP authentication is VRF aware.
19.1R1	Starting in Junos OS Release 19.1R1, Junos OS extends support for TCP authentication to BGP peers that are discovered through allowed prefix subnets configured in a BGP group.

IP Security for BGP

IN THIS SECTION

- Understanding IPsec for BGP | 1245
- Example: Using IPsec to Protect BGP Traffic | 1246

Understanding IPsec for BGP

You can apply the IP security (IPsec) to BGP traffic. IPsec is a protocol suite used for protecting IP traffic at the packet level. IPsec is based on security associations (SAs). An SA is a simplex connection that provides security services to the packets carried by the SA. After configuring the SA, you can apply it to BGP peers.

The Junos OS implementation of IPsec supports two types of security: host to host and gateway to gateway. Host-to-host security protects BGP sessions with other routers. An SA to be used with BGP must be configured manually and use transport mode. Static values must be configured on both ends of

the security association. To apply host protection, you configure manual SAs in transport mode and then reference the SA by name in the BGP configuration to protect a session with a given peer.

Manual SAs require no negotiation between the peers. All values, including the keys, are static and specified in the configuration. Manual SAs statically define the security parameter index values, algorithms, and keys to be used and require matching configurations on both end points of the tunnel (on both peers). As a result, each peer must have the same configured options for communication to take place.

In transport mode, IPsec headers are inserted after the original IP header and before the transport header.

The security parameter index is an arbitrary value used in combination with a destination address and a security protocol to uniquely identify the SA.

SEE ALSO

Understanding Router Authentication for BGP | 1234

Example: Using IPsec to Protect BGP Traffic

IN THIS SECTION

- Requirements | 1246
- Overview | 1247
- Configuration | 1248
- Verification | 1250

IPsec is a suite of protocols used to provide secure network connections at the IP layer. It is used to provide data source authentication, data integrity, confidentiality and packet replay protection. This example shows how to configure IPsec functionality to protect Routing Engine-to-Routing Engine BGP sessions. Junos OS supports IPsec Authentication Header (AH) and Encapsulating Security Payload (ESP) in transport and tunnel mode, as well as a utility for creating policies and manually configuring keys.

Requirements

Before you begin:

- Configure the router interfaces.
- Configure an interior gateway protocol (IGP).
- Configure BGP.

No specific PIC hardware is required to configure this feature.

Overview

IN THIS SECTION

- Topology Diagram | **1247**

The SA is configured at the `[edit security ipsec security-association name]` hierarchy level with the `mode` statement set to transport. In transport mode, Junos OS does not support authentication header (AH) or encapsulating security payload (ESP) header bundles. Junos OS supports only the BGP protocol in transport mode.

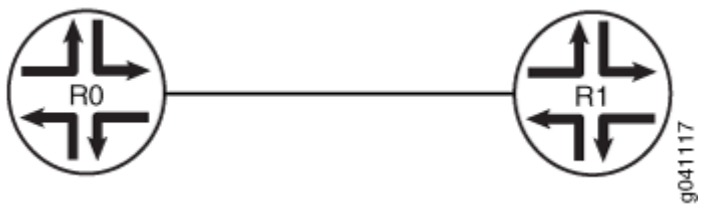
This example specifies bidirectional IPsec to decrypt and authenticate the incoming and outgoing traffic using the same algorithm, keys, and SPI in both directions, unlike inbound and outbound SAs that use different attributes in both directions.

A more specific SA overrides a more general SA. For example, if a specific SA is applied to a specific peer, that SA overrides the SA applied to the whole peer group.

Topology Diagram

Figure 84 on page 1247 shows the topology used in this example.

Figure 84: IPsec for BGP



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1248
- Procedure | 1248

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.

```
[edit]
set security ipsec security-association test-sa mode transport
set security ipsec security-association test-sa manual direction bidirectional protocol esp
set security ipsec security-association test-sa manual direction bidirectional spi 1000
set security ipsec security-association test-sa manual direction bidirectional encryption
algorithm 3des-cbc
set security ipsec security-association test-sa manual direction bidirectional encryption key
ascii-text "$9$kPT3AtO1hr6/u1IhvM8X7Vb2JGimfz.PtuB1hcs2goGDkqf5Qndb.5QzCA0BIRrvx7VsgJ"
set protocols bgp group 1 neighbor 10.1.1.1 ipsec-sa test-sa
```

Procedure

Step-by-Step Procedure

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

To configure Router R1:

1. Configure the SA mode.

```
[edit security ipsec security-association test-sa]
user@R1# set mode transport
```

2. Configure the IPsec protocol to be used.

```
[edit security ipsec security-association test-sa]
user@R1# set manual direction bidirectional protocol esp
```

3. Configure to security parameter index to uniquely identify the SA.

```
[edit security ipsec security-association test-sa]
user@R1# set manual direction bidirectional spi 1000
```

4. Configure the encryption algorithm.

```
[edit security ipsec security-association test-sa]
user@R1# set manual direction bidirectional encryption algorithm 3des-cbc
```

5. Configure the encryption key.

```
[edit security ipsec security-association test-sa]
user@R1# set manual direction bidirectional encryption key ascii-text "$9$kPT3AtO1hr6/
u1IhvM8X7Vb2JGimfz.PtuB1hcs2goGDkqf5Qndb.5QzCA0BIRrvx7VsgJ"
```

When you use an ASCII text key, the key must contain exactly 24 characters.

6. Apply the SA to the BGP peer.

```
[edit protocols bgp group 1 neighbor 10.1.1.1]
user@R1# set ipsec-sa test-sa
```

Results

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

```
user@R1# show protocols
bgp {
    group 1 {
```

```
        neighbor 10.1.1.1 {
            ipsec-sa test-sa;
        }
    }
}
```

```
user@R1# show security
ipsec {
    security-association test-sa {
        mode transport;
        manual {
            direction bidirectional {
                protocol esp;
                spi 1000;
                encryption {
                    algorithm 3des-cbc;
                    key ascii-text "$9$kPT3AtO1hr6/
u1IhvM8X7Vb2JGimfz.PtuB1hcs2goGDkqf5Qndb.5QzCA0BIRrvx7VsgJ"; ## SECRET-DATA
                }
            }
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode. Repeat the configuration on Router R0, changing only the neighbor address.

Verification

IN THIS SECTION

- Verifying the Security Associaton | 1251

Confirm that the configuration is working properly.

Verifying the Security Associaton

Purpose

Make sure that the correct settings appear in the output of the `show ipsec security-associations` command.

Action

From operational mode, enter the `show ipsec security-associations` command.

```
user@R1> show ipsec security-associations
Security association: test-sa
    Direction SPI         AUX-SPI     Mode       Type     Protocol
    inbound   1000        0           transport  manual   ESP
    outbound  1000        0           transport  manual   ESP
```

Meaning

The output is straighforward for most fields except the AUX-SPI field. The AUX-SPI is the value of the auxiliary security parameter index. When the value is AH or ESP, AUX-SPI is always 0. When the value is AH+ESP, AUX-SPI is always a positive integer.

SEE ALSO

Understanding IPsec for BGP | 1245

TCP Access Restriction for BGP

IN THIS SECTION

- Understanding Security Options for BGP with TCP | 1252
- Example: Configuring a Filter to Block TCP Access to a Port Except from Specified BGP Peers | 1252
- Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List | 1261
- Example: Limiting TCP Segment Size for BGP | 1266

Understanding Security Options for BGP with TCP

Among routing protocols, BGP is unique in using TCP as its transport protocol. BGP peers are established by manual configuration between routing devices to create a TCP session on port 179. A BGP-enabled device periodically sends keepalive messages to maintain the connection.

Over time, BGP has become the dominant interdomain routing protocol on the Internet. However, it has limited guarantees of stability and security. Configuring security options for BGP must balance suitable security measures with acceptable costs. No one method has emerged as superior to other methods. Each network administrator must configure security measures that meet the needs of the network being used.

For detailed information about the security issues associated with BGP's use of TCP as a transport protocol, see RFC 4272, *BGP Security Vulnerabilities Analysis*.

SEE ALSO

Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List | **1261**

Example: Limiting TCP Segment Size for BGP | **1266**

Example: Configuring a Filter to Block TCP Access to a Port Except from Specified BGP Peers

IN THIS SECTION

- Requirements | **1253**
- Overview | **1253**
- Configuration | **1254**
- Verification | **1258**

This example shows how to configure a standard stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except from specified BGP peers.

Requirements

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

Overview

IN THIS SECTION

- Topology | **1253**

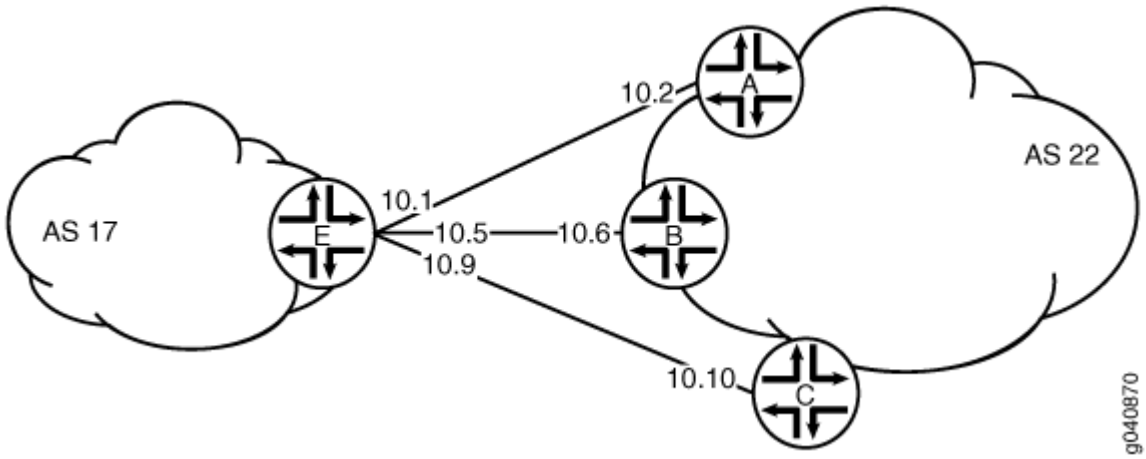
In this example, you create a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except the specified BGP peers.

The stateless firewall filter **filter_bgp179** matches all packets from the directly connected interfaces on Device A and Device B to the destination port number 179.

Topology

Figure 85 on page 1253 shows the topology used in this example. Device C attempts to make a TCP connection to Device E. Device E blocks the connection attempt. This example shows the configuration on Device E.

Figure 85: Typical Network with BGP Peer Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1254
- Configuring Device E | 1255

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 C

```
set interfaces ge-1/2/0 unit 10 description to-E
set interfaces ge-1/2/0 unit 10 family inet address 10.10.10.10/30
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 17
set protocols bgp group external-peers neighbor 10.10.10.9
set routing-options autonomous-system 22
```

Device E

```
set interfaces ge-1/2/0 unit 0 description to-A
set interfaces ge-1/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/2/1 unit 5 description to-B
set interfaces ge-1/2/1 unit 5 family inet address 10.10.10.5/30
set interfaces ge-1/0/0 unit 9 description to-C
set interfaces ge-1/0/0 unit 9 family inet address 10.10.10.9/30
set interfaces lo0 unit 2 family inet filter input filter_bgp179
set interfaces lo0 unit 2 family inet address 192.168.0.1/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 22
set protocols bgp group external-peers neighbor 10.10.10.2
set protocols bgp group external-peers neighbor 10.10.10.6
set protocols bgp group external-peers neighbor 10.10.10.10
set routing-options autonomous-system 17
```

```
set firewall family inet filter filter_bgp179 term 1 from source-address 10.10.10.2/32
set firewall family inet filter filter_bgp179 term 1 from source-address 10.10.10.6/32
set firewall family inet filter filter_bgp179 term 1 from destination-port bgp
set firewall family inet filter filter_bgp179 term 1 then accept
set firewall family inet filter filter_bgp179 term 2 then reject
```

Configuring Device E

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 Junos OS CLI User Guide.

To configure Device E with a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requestors except specified BGP peers:

1. Configure the interfaces.

```
user@E# set interfaces ge-1/2/0 unit 0 description to-A
user@E# set interfaces ge-1/2/0 unit 0 family inet address 10.10.10.1/30
user@E# set interfaces ge-1/2/1 unit 5 description to-B
user@E# set interfaces ge-1/2/1 unit 5 family inet address 10.10.10.5/30
user@E# set interfaces ge-1/0/0 unit 9 description to-C
user@E# set interfaces ge-1/0/0 unit 9 family inet address 10.10.10.9/30
```

2. Configure BGP.

```
[edit protocols bgp group external-peers]
user@E# set type external
user@E# set peer-as 22
user@E# set neighbor 10.10.10.2
user@E# set neighbor 10.10.10.6
user@E# set neighbor 10.10.10.10
```

3. Configure the autonomous system number.

```
[edit routing-options]
user@E# set autonomous-system 17
```

4. Define the filter term that accepts TCP connection attempts to port 179 from the specified BGP peers.

```
[edit firewall family inet filter filter_bgp179]
user@E# set term 1 from source-address 10.10.10.2/32
user@E# set term 1 from source-address 10.10.10.6/32
user@E# set term 1 from destination-port bgp
user@E# set term 1 then accept
```

5. Define the other filter term to reject packets from other sources.

```
[edit firewall family inet filter filter_bgp179]
user@E# set term 2 then reject
```

6. Apply the firewall filter to the loopback interface.

```
[edit interfaces lo0 unit 2 family inet]
user@E# set filter input filter_bgp179
user@E# set address 192.168.0.1/32
```

Results

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

```
user@E# show firewall
family inet {
    filter filter_bgp179 {
        term 1 {
            from {
                source-address {
                    10.10.10.2/32;
                    10.10.10.6/32;
                }
                destination-port bgp;
            }
            then accept;
        }
```

```
        term 2 {
            then {
                reject;
            }
        }
    }
}
```

```
user@E# show interfaces
lo0 {
    unit 2 {
        family inet {
            filter {
                input filter_bgp179;
            }
            address 192.168.0.1/32;
        }
    }
}
ge-1/2/0 {
    unit 0 {
        description to-A;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
ge-1/2/1 {
    unit 5 {
        description to-B;
        family inet {
            address 10.10.10.5/30;
        }
    }
}
ge-1/0/0 {
    unit 9 {
        description to-C;
        family inet {
            address 10.10.10.9/30;
        }
```

```
    }
}
```

```
user@E# show protocols
bgp {
    group external-peers {
        type external;
        peer-as 22;
        neighbor 10.10.10.2;
        neighbor 10.10.10.6;
        neighbor 10.10.10.10;
    }
}
```

```
user@E# show routing-options
autonomous-system 17;
```

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

Verification

IN THIS SECTION

- Verifying That the Filter Is Configured | **1258**
- Verifying the TCP Connections | **1259**
- Monitoring Traffic on the Interfaces | **1259**

Confirm that the configuration is working properly.

Verifying That the Filter Is Configured

Purpose

Make sure that the filter is listed in output of the `show firewall filter` command.

Action

```
user@E> show firewall filter filter_bgp179
Filter: filter_bgp179
```

Verifying the TCP Connections

Purpose

Verify the TCP connections.

Action

From operational mode, run the `show system connections extensive` command on Device C and Device E.

The output on Device C shows the attempt to establish a TCP connection. The output on Device E shows that connections are established with Device A and Device B only.

```
user@C> show system connections extensive | match 10.10.10

tcp4       0      0  10.10.10.9.51872     10.10.10.10.179       SYN_SENT
```

```
user@E> show system connections extensive | match 10.10.10

tcp4       0      0  10.10.10.5.179        10.10.10.6.62096     ESTABLISHED
tcp4       0      0  10.10.10.6.62096      10.10.10.5.179       ESTABLISHED
tcp4       0      0  10.10.10.1.179        10.10.10.2.61506     ESTABLISHED
tcp4       0      0  10.10.10.2.61506      10.10.10.1.179       ESTABLISHED
```

Monitoring Traffic on the Interfaces

Purpose

Use the **monitor traffic** command to compare the traffic on an interface that establishes a TCP connection with the traffic on an interface that does not establish a TCP connection.

Action

From operational mode, run the **monitor traffic** command on the Device E interface to Device B and on the Device E interface to Device C. The following sample output verifies that in the first example, acknowledgment (**ack**) messages are received. In the second example, **ack** messages are not received.

```
user@E> monitor traffic size 1500 interface ge-1/2/1.5
19:02:49.700912 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: P 3330573561:3330573580(19) ack
915601686 win 16384 <nop,nop,timestamp 1869518816 1869504850>: BGP, length: 19
19:02:49.801244  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: . ack 19 win 16384 <nop,nop,timestamp
1869518916 1869518816>
19:03:03.323018  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: P 1:20(19) ack 19 win 16384
<nop,nop,timestamp 1869532439 1869518816>: BGP, length: 19
19:03:03.422418 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: . ack 20 win 16384 <nop,nop,timestamp
1869532539 1869532439>
19:03:17.220162 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: P 19:38(19) ack 20 win 16384
<nop,nop,timestamp 1869546338 1869532439>: BGP, length: 19
19:03:17.320501  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: . ack 38 win 16384 <nop,nop,timestamp
1869546438 1869546338>
```

```
user@E> monitor traffic size 1500 interface ge-1/0/0.9

18:54:20.175471 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,nop,wscale 0,nop,nop,timestamp 1869009240 0,sackOK,eol>
18:54:23.174422 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,nop,wscale 0,nop,nop,timestamp 1869012240 0,sackOK,eol>
18:54:26.374118 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,nop,wscale 0,nop,nop,timestamp 1869015440 0,sackOK,eol>
18:54:29.573799 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,sackOK,eol>
18:54:32.773493 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,sackOK,eol>
18:54:35.973185 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0) win 16384
<mss 1460,sackOK,eol>
```

SEE ALSO

Understanding How to Use Standard Firewall Filters

Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods

Example: Configuring a Filter to Accept Packets Based on IPv6 TCP Flags

Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List

IN THIS SECTION

- Requirements | **1261**
- Overview | **1261**
- Configuration | **1262**
- Verification | **1265**

This example shows how to configure a standard stateless firewall filter that limits certain TCP and Internet Control Message Protocol (ICMP) traffic destined for the Routing Engine by specifying a list of prefix sources that contain allowed BGP peers.

Requirements

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

Overview

IN THIS SECTION

- Topology | **1261**

In this example, you create a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except BGP peers that have a specified prefix.

Topology

A source prefix list, **plist_bgp179**, is created that specifies the list of source prefixes that contain allowed BGP peers.

The stateless firewall filter **filter_bgp179** matches all packets from the source prefix list **plist_bgp179** to the destination port number 179.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1262**
- Configure the Filter | **1262**
- Results | **1263**

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.

```
set policy-options prefix-list plist_bgp179 apply-path "protocols bgp group <*> neighbor <*>"
set firewall family inet filter filter_bgp179 term 1 from source-address 0.0.0.0/0
set firewall family inet filter filter_bgp179 term 1 from source-prefix-list plist_bgp179 except
set firewall family inet filter filter_bgp179 term 1 from destination-port bgp
set firewall family inet filter filter_bgp179 term 1 then reject
set firewall family inet filter filter_bgp179 term 2 then accept
set interfaces lo0 unit 0 family inet filter input filter_bgp179
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Configure the Filter

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 Junos OS CLI User Guide.

To configure the filter:

1. Expand the prefix list **bgp179** to include all prefixes pointed to by the BGP peer group defined by **protocols bgp group <*> neighbor <*>**.

```
[edit policy-options prefix-list plist_bgp179]
user@host# set apply-path " protocols bgp group <*> neighbor <*>"
```

2. Define the filter term that rejects TCP connection attempts to port 179 from all requesters except the specified BGP peers.

```
[edit firewall family inet filter filter_bgp179]
user@host# set term term1 from source-address 0.0.0.0/0
user@host# set term term1 from source-prefix-list bgp179 except
user@host# set term term1 from destination-port bgp
user@host# set term term1 then reject
```

3. Define the other filter term to accept all packets.

```
[edit firewall family inet filter filter_bgp179]
user@host# set term term2 then accept
```

4. Apply the firewall filter to the loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@host# set filter input filter_bgp179
user@host# set address 127.0.0.1/32
```

Results

From configuration mode, confirm your configuration by entering the **show firewall**, **show interfaces**, 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@host# show firewall
family inet {
    filter filter_bgp179 {
        term 1 {
            from {
                source-address {
```

```
                    0.0.0.0/0;
                }
                source-prefix-list {
                    plist_bgp179 except;
                }
                destination-port bgp;
            }
            then {
                reject;
            }
        }
        term 2 {
            then {
                accept;
            }
        }
    }
}
```

```
user@host# show interfaces
lo0 {
    unit 0 {
        family inet {
            filter {
                input filter_bgp179;
            }
            address 127.0.0.1/32;
        }
    }
}
```

```
user@host# show policy-options
prefix-list plist_bgp179 {
    apply-path "protocols bgp group <*> neighbor <*>";
}
```

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

Verification

IN THIS SECTION

- Displaying the Firewall Filter Applied to the Loopback Interface | **1265**

Confirm that the configuration is working properly.

Displaying the Firewall Filter Applied to the Loopback Interface

Purpose

Verify that the firewall filter **filter_bgp179** is applied to the IPv4 input traffic at logical interface **lo0.0**.

Action

Use the `show interfaces statistics operational mode` command for logical interface **lo0.0**, and include the **detail** option. Under the **Protocol inet** section of the command output section, the **Input Filters** field displays the name of the stateless firewall filter applied to the logical interface in the input direction.

```
[edit]
user@host> show interfaces statistics lo0.0 detail
  Logical interface lo0.0 (Index 321) (SNMP ifIndex 16) (Generation 130)
    Flags: SNMP-Traps Encapsulation: Unspecified
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Local statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Transit statistics:
     Input  bytes  :                   0                 0 bps
     Output bytes  :                   0                 0 bps
     Input  packets:                   0                 0 pps
```

```
 Output packets:                    0                    0 pps
Protocol inet, MTU: Unlimited, Generation: 145, Route table: 0
  Flags: Sendbcast-pkt-to-re
  Input Filters: filter_bgp179
  Addresses, Flags: Primary
    Destination: Unspecified, Local: 127.0.0.1, Broadcast: Unspecified, Generation: 138
```

SEE ALSO

- *Understanding How to Use Standard Firewall Filters*
- *Firewall Filter Match Conditions Based on Address Fields*
- *Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods*
- *Example: Configuring a Filter to Accept Packets Based on IPv6 TCP Flags*
- *prefix-list*

Example: Limiting TCP Segment Size for BGP

IN THIS SECTION

- Requirements | **1266**
- Overview | **1267**
- Configuration | **1268**
- Verification | **1271**
- Troubleshooting | **1271**

This example shows how to avoid Internet Control Message Protocol (ICMP) vulnerability issues by limiting TCP segment size when you are using maximum transmission unit (MTU) discovery. Using MTU discovery on TCP paths is one method of avoiding BGP packet fragmentation.

Requirements

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

Overview

IN THIS SECTION

- Topology Diagram | **1267**

TCP negotiates a maximum segment size (MSS) value during session connection establishment between two peers. The MSS value negotiated is primarily based on the maximum transmission unit (MTU) of the interfaces to which the communicating peers are directly connected. However, due to variations in link MTU on the path taken by the TCP packets, some packets in the network that are well within the MSS value might be fragmented when the packet size exceeds the link's MTU.

To configure the TCP MSS value, include the `tcp-mss` statement with a segment size from 1 through 4096.

If the router receives a TCP packet with the SYN bit and the MSS option set, and the MSS option specified in the packet is larger than the MSS value specified by the `tcp-mss` statement, the router replaces the MSS value in the packet with the lower value specified by the `tcp-mss` statement.

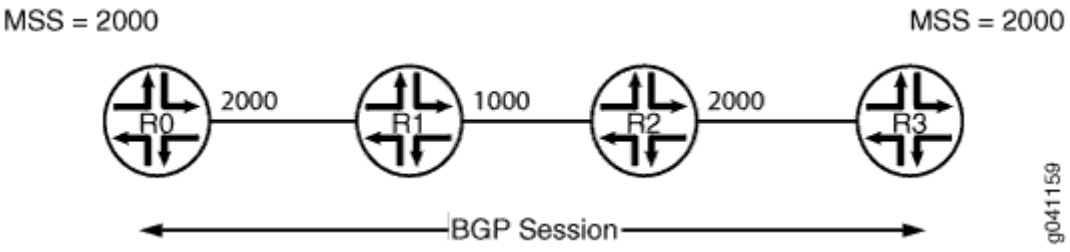
The configured MSS value is used as the maximum segment size for the sender. The assumption is that the TCP MSS value used by the sender to communicate with the BGP neighbor is the same as the TCP MSS value that the sender can accept from the BGP neighbor. If the MSS value from the BGP neighbor is less than the MSS value configured, the MSS value from the BGP neighbor is used as the maximum segment size for the sender.

This feature is supported with TCP over IPv4 and TCP over IPv6.

Topology Diagram

Figure 86 on page 1267 shows the topology used in this example.

Figure 86: TCP Maximum Segment Size for BGP



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1268
- Procedure | 1268

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.

R0

```
set interfaces fe-1/2/0 unit 1 family inet address 1.1.0.1/30
set interfaces lo0 unit 1 family inet address 10.255.14.179/32
set protocols bgp group-int tcp-mss 2020
set protocols bgp group int type internal
set protocols bgp group int local-address 10.255.14.179
set protocols bgp group int mtu-discovery
set protocols bgp group int neighbor 10.255.71.24 tcp-mss 2000
set protocols bgp group int neighbor 10.255.14.177
set protocols bgp group int neighbor 10.0.14.4 tcp-mss 4000
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface 10.255.14.179
set routing-options autonomous-system 65000
```

Procedure

Step-by-Step Procedure

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

To configure Router R0:

1. Configure the interfaces.

```
[edit interfaces]
user@R0# set fe-1/2/0 unit 1 family inet address 1.1.0.1/30
user@R0# set lo0 unit 1 family inet address 10.255.14.179/32
```

2. Configure an interior gateway protocol (IGP), OSPF in this example.

```
[edit protocols ospf area 0.0.0.0]
user@R0# set interface fe-1/2/0.1
user@R0# set interface 10.255.14.179
```

3. Configure one or more BGP groups.

```
[edit protocols bgp group int]
user@R0# set type internal
user@R0# set local-address 10.255.14.179
```

4. Configure MTU discovery to prevent packet fragmentation.

```
[edit protocols bgp group int]
user@R0# set mtu-discovery
```

5. Configure the BGP neighbors, with the TCP MSS set globally for the group or specifically for the various neighbors.

```
[edit protocols bgo group int]
user@R0# set tcp-mss 2020
user@R0# set neighbor 10.255.14.177
user@R0# set neighbor 10.255.71.24 tcp-mss 2000
user@R0# set neighbor 10.0.14.4 tcp-mss 4000
```

NOTE: The TCP MSS neighbor setting overrides the group setting.

6. Configure the local autonomous system.

```
[edit routing-options]
user@R0# set autonomous-system 65000
```

Results

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

```
user@R0# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 1.1.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.255.14.179/32;
        }
    }
}
```

```
user@R0# show protocols
bgp {
    group int {
        type internal;
        local-address 10.255.14.179;
        mtu-discovery;
        tcp-mss 2020;
        neighbor 10.255.71.24 {
            tcp-mss 2000;
        }
        neighbor 10.255.14.177;
        neighbor 10.0.14.4 {
```

```
            tcp-mss 4000;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.1;
        interface 10.255.14.179;
    }
}
```

```
user@R0# show routing-options
autonomous-system 65000;
```

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

Verification

To confirm that the configuration is working properly, run the following commands:

- `show system connections extensive | find <neighbor-address>`, to check the negotiated TCP MSS value.
- `monitor traffic interface`, to monitor BGP traffic and to make sure that the configured TCP MSS value is used as the MSS option in the TCP SYN packet.

Troubleshooting

IN THIS SECTION

- MSS Calculation with MTU Discovery | **1272**

MSS Calculation with MTU Discovery

Problem

Consider an example in which two routing devices (R1 and R2) have an internal BGP (IBGP) connection. On both of the routers, the connected interfaces have 4034 as the IPv4 MTU.

```
user@R1# show protocols bgp | display set
[edit]
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 45.45.45.2
set protocols bgp group ibgp mtu-discovery
set protocols bgp group ibgp neighbor 45.45.45.1
```

```
user@R1# run show interfaces xe-0/0/3 extensive | match mtu

  Link-level type: Ethernet, MTU: 4048, LAN-PHY mode, Speed: 10Gbps,
    FIFO errors: 0, HS link CRC errors: 0, MTU errors: 0, Resource errors: 0
    Protocol inet, MTU: 4034, Generation: 180, Route table: 0
    Protocol multiservice, MTU: Unlimited, Generation: 181, Route table: 0
```

In the following packet capture on Device R1, the negotiated MSS is 3994. In the `show system connections extensive` information for MSS, it is set to 2048.

```
05:50:01.575218 Out
        Juniper PCAP Flags [Ext], PCAP Extension(s) total length 16
          Device Media Type Extension TLV #3, length 1, value: Ethernet (1)
          Logical Interface Encapsulation Extension TLV #6, length 1, value: Ethernet (14)
          Device Interface Index Extension TLV #1, length 2, value: 137
          Logical Interface Index Extension TLV #4, length 4, value: 69
        -----original packet-----
        00:21:59:e1:e8:03 > 00:19:e2:20:79:01, ethertype IPv4 (0x0800), length 78: (tos 0xc0,
ttl 64, id 53193, offset 0, flags [DF], proto: TCP (6), length: 64) 45.45.45.2.62840 >
45.45.45.1.bgp: S 2939345813:2939345813(0) win 16384 **mss 3994,nop,wscale 0,nop,nop,timestamp
70559970 0,sackOK,eol>
05:50:01.575875 In
        Juniper PCAP Flags [Ext, no-L2, In], PCAP Extension(s) total length 16
          Device Media Type Extension TLV #3, length 1, value: Ethernet (1)
          Logical Interface Encapsulation Extension TLV #6, length 1, value: Ethernet
```

```
(14)
          Device Interface Index Extension TLV #1, length 2, value: 137
          Logical Interface Index Extension TLV #4, length 4, value: 69
        -----original packet-----
        PFE proto 2 (ipv4): (tos 0xc0, ttl 255, id 37709, offset 0, flags [DF], proto: TCP (6),
length: 64) 45.45.45.1.bgp > 45.45.45.2.62840: S 2634967984:2634967984(0) ack 2939345814 win
16384 **mss 3994,nop,wscale 0,nop,nop,timestamp 174167273 70559970,sackOK,eol>
```

```
user@R1# run show system connections extensive | find 45.45

tcp4       0      0  45.45.45.2.62840
45.45.45.1.179                             ESTABLISHED
   sndsbcc:          0 sndsbmbcnt:          0  sndsbmbmax:     131072
sndsblowat:       2048 sndsbhiwat:      16384
   rcvsbcc:          0 rcvsbmbcnt:          0  rcvsbmbmax:     131072
rcvsblowat:          1 rcvsbhiwat:      16384
   proc id:      19725  proc name:        rpd
       iss: 2939345813      sndup: 2939345972
    snduna: 2939345991     sndnxt: 2939345991      sndwnd:      16384
    sndmax: 2939345991    sndcwnd:      10240 sndssthresh: 1073725440
       irs: 2634967984      rcvup: 2634968162
    rcvnxt: 2634968162     rcvadv: 2634984546      rcvwnd:      16384
       rtt:          0       srtt:       1538        rttv:       1040
    rxtcur:       1200   rxtshift:          0       rtseq: 2939345972
    rttmin:       1000  mss:        2048
```

Solution

This is expected behavior with Junos OS. The MSS value is equal to the MTU value minus the IP or IPv6 and TCP headers. This means that the MSS value is generally 40 bytes less than the MTU (for IPv4) and 60 bytes less than the MTU (for IPv6). This value is negotiated between the peers. In this example, it is 4034 - 40 = 3994. Junos OS then rounds this value to a multiple of 2 KB. The value is 3994 / 2048 * 2048=2048. So it is not necessary to see same MSS value with in the `show system connections` output.

3994 / 2048 = 1.95

1.95 is rounded to 1.

1 * 2048 = 2048

SEE ALSO

BGP Configuration Overview | 24

Understanding External BGP Peering Sessions | 26

BGP Origin Validation

IN THIS SECTION

- Understanding Origin Validation for BGP | 1274
- Use Case and Benefit of Origin Validation for BGP | 1282
- Example: Configuring Origin Validation for BGP | 1283

Understanding Origin Validation for BGP

IN THIS SECTION

- Supported Standards | 1275
- How Origin Validation Works | 1276
- BGP Interaction with the Route Validation Database | 1278
- Community Attribute to Announce RPKI Validation State to IBGP Neighbors | 1280
- Nonstop Active Routing and Origin Validation | 1281
- Marking a Prefix Range as Never Allowed | 1282

Origin validation helps to prevent the unintentional advertisement of routes. Sometimes network administrators mistakenly advertise routes to networks that they do not control. You can resolve this security issue by configuring origin validation (also known as secure interdomain routing). Origin validation is a mechanism by which route advertisements can be authenticated as originating from an expected autonomous system (AS). Origin validation uses one or more resource public key infrastructure (RPKI) cache servers to perform authentication for specified BGP prefixes. To authenticate a prefix, the

router (BGP speaker) queries the database of validated prefix-to-AS mappings, which are downloaded from the cache server, and ensures that the prefix originated from an expected AS.

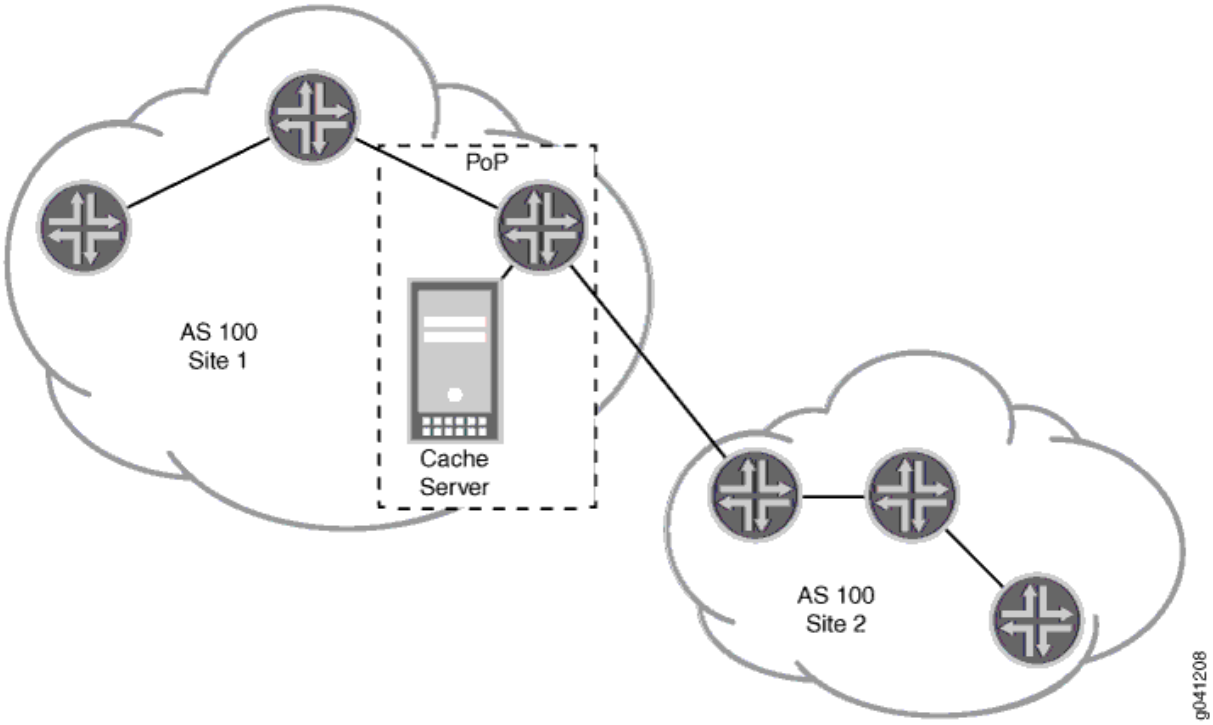
NOTE: Prior to Junos OS Releases 20.1R3, 20.2R3, 20.3R2, 20.4R2, 21.1R1, when you enable the RPKI authentication, Junos OS opens the TCP port 2222 automatically without any notice. You can apply a filter to block and secure this port.

Starting in Junos OS Releases 20.1R3, 20.2R3, 20.3R2, 20.4R2, 21.1R1, when you enable the RPKI authentication, Junos OS no longer opens the TCP ports 2222 automatically.

Junos OS supports origin validation for IPv4 and IPv6 prefixes.

Figure 87 on page 1275 shows a sample topology.

Figure 87: Sample Topology for Origin Validation



Supported Standards

The Junos OS implementation of origin validation supports the following RFCs and draft:

- RFC 6810, *The Resource Public Key Infrastructure (RPKI) to Router Protocol*

- RFC 6811, *BGP Prefix Origin Validation*
- Internet draft draft-ietf-sidr-origin-validation-signaling-00, *BGP Prefix Origin Validation State Extended Community* (partial support)

  The extended community (origin validation state) is supported in Junos OS routing policy. The specified change in the route selection procedure is not supported.

How Origin Validation Works

The RPKI and origin validation use X.509 certificates with extensions specified in RFC 3779, *X.509 Extensions for IP Addresses and AS Identifiers*.

The RPKI consists of a distributed collection of information. Each Certification Authority publishes its end-entity (EE) certificates, certificate revocation lists (CRLs), and signed objects at a particular location. All of these repositories form a complete set of information that is available to every RPKI cache server.

Each RPKI cache server maintains a local cache of the entire distributed repository collection by regularly synchronizing each element in the local cache against the original repository publication point.

On the router, the database entries are formatted as *route validation (RV) records*. An RV record is a (prefix, maximum length, origin AS) triple. It matches any route whose prefix matches the RV prefix, whose prefix length does not exceed the maximum length given in the RV record, and whose origin AS equals the origin AS given in the RV record.

An RV record is a simplified version of a *route origin authorization (ROA)*. An ROA is a digitally signed object that provides a means of verifying that an IP address block holder has authorized an AS to originate routes to one or more prefixes within the address block. ROAs are not directly used in route validation. The cache server exports a simplified version of the ROA to the router as an RV record.

The maximum length value must be greater than or equal to the length of the authorized prefix and less than or equal to the length (in bits) of an IP address in the address family (32 for IPv4 and 128 for IPv6). The maximum length defines the IP address prefix that the AS is authorized to advertise.

For example, if the IP address prefix is 200.4.66/24, and the maximum length is 26, the AS is authorized to advertise 200.4.66.0/24, 200.4.66.0/25, 200.4.66.128/25, 200.4.66.0/26, 200.4.66.64/26, 200.4.66.128/26, and 200.4.66.192/26. When the maximum length is not present, the AS is only authorized to advertise exactly the prefix specified in the RV.

As another example, an RV can contain the prefix 200.4.66/24 with a maximum length of 26, as well as the prefix 200.4.66.0/28 with a maximum length of 28. This RV would authorize the AS to advertise any prefix beginning with 200.4.66 with a length of at least 24 and no greater than 26, as well as the specific prefix 200.4.66.0/28.

The origin of a route is represented by the right-most AS number in the AS_PATH attribute. Origin validation operates by comparing the origin AS in a routing update with the authorized source AS published in RV records.

The security provided by origin validation alone is known to be weak against a determined attacker because there is no protection against such an attacker spoofing the source AS. That said, origin validation provides useful protection against accidental announcements.

Although origin validation could be implemented by having each router directly participate in the RPKI, this is seen as too resource intensive (because many public-key cryptography operations are required to validate the RPKI data) as well as operationally intensive to set up and maintain an RPKI configuration on each router. For this reason, a separate RPKI cache server performs public-key validations, and generates a validated database of prefix-to-AS mappings. The validated database is downloaded to a client router over a secure TCP connection. The router thus requires little information about the RPKI infrastructure and has no public-key cryptography requirements, other than the encrypted transport password. The router subsequently uses the downloaded data to validate received route updates.

When you configure server sessions, you can group the sessions together and configure session parameters for each session in the group. The router tries periodically to set up a configurable maximum number of connections to cache servers. If connection setup fails, a new connection attempt is made periodically.

In the meantime, after the validation import policy is applied to the BGP session, route-validation is performed irrespective of cache session state (up or down) and RV database (empty or not empty). If the RV database is empty or none of the cache server sessions are up, the validation state for each route is set to unknown, because no RV record exists to evaluate a received BGP prefix.

The retry-attempt period is configurable. After successfully connecting to a cache server, the router queries for the latest database serial number and requests that the RPKI cache transmits all of the RV entries belonging to that version of the database.

Each inbound message resets a liveliness timer for the RPKI cache server. After all updates are learned, the router performs periodic liveliness checks based on a configurable interval. This is done by sending a serial query protocol data unit (PDU) with the same serial number that the cache server reported in its latest notification PDU. The cache server responds with zero or more updates and an end-of-data (EOD) PDU, which also refreshes the liveliness state of the cache server and resets a record-lifetime timer.

When a prefix is received from an external BGP (EBGP) peer, it is examined by an import policy and marked as Valid, Invalid, Unknown, or Unverified:

- Valid—Indicates that the prefix and AS pair are found in the database.
- Invalid—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database.
- Unknown—Indicates that the prefix is not among the prefixes or prefix ranges in the database.

- Unverified—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers.

If there are any potential matches for the route in the validation database, the route has to match one of them to be valid. Otherwise, it is invalid. Any match is adequate to make the route valid. It does not need to be a best match. Only if there are no potential matches is the route considered to be unknown. For more information about the prefix-to-AS mapping database logic, see Section 2 of Internet draft draft-ietf-sidr-pfx-validate-01, *BGP Prefix Origin Validation*.

NOTE: RPKI validation is available only in the primary instance. If you configure RPKI validation for a routing instance, then the RPKI validation fails with the following error message `RV instance is not running`.

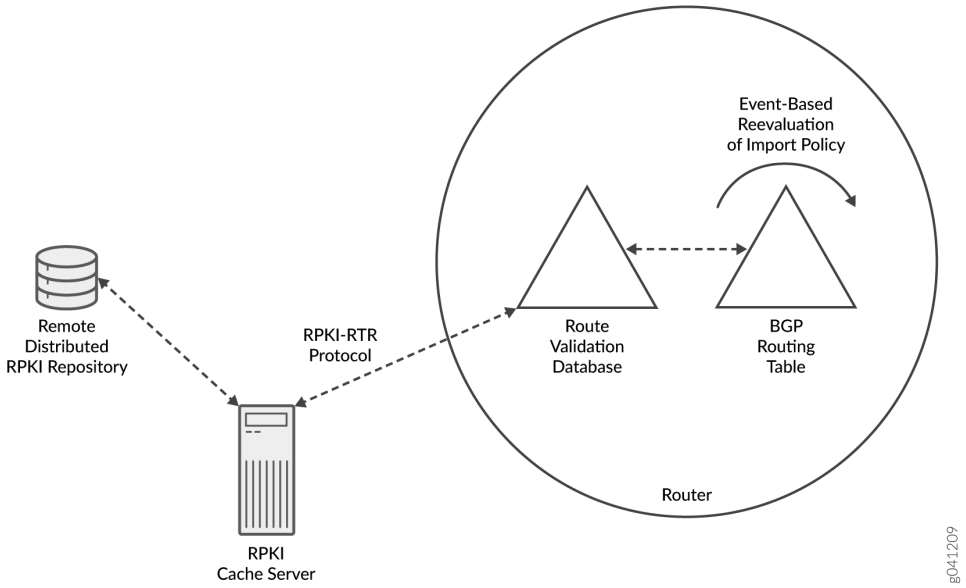
BGP Interaction with the Route Validation Database

The route validation (RV) database contains a collection of RV records that the router downloads from the RPKI cache server. After the RV database is populated with RV records, the RV database scans the RIB-Local routing table to determine if there are any prefixes in RIB-Local that might be affected by the RV records in the database. (RIB-Local contains the IPv4 and IPv6 routes shown in the output of the `show route protocol bgp` command.)

This process triggers a BGP reevaluation of BGP import policies (not export policies).

Figure 88 on page 1279 shows the process.

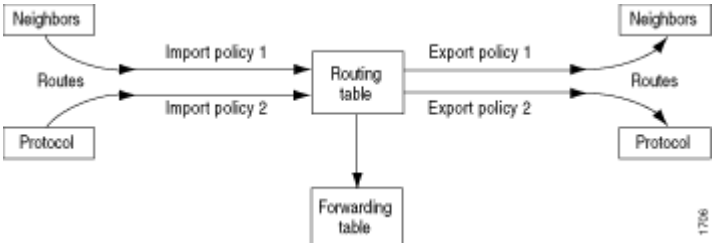
Figure 88: BGP and Route Validation



Import policies are applied to RIB-In. Another way to understand this is that Import policies are applied to the routes that are shown in the output of the `show route receive-protocol bgp` command, while export policies are applied to routes that are shown by the `show route advertising-protocol bgp` command.

As shown in Figure 89 on page 1279, you use import routing policies to control which routes BGP places in the routing table, and export routing policies to control which routes BGP advertises from the routing table to its neighbors.

Figure 89: Importing and Exporting Routing Policies



When you configure a route-validation import policy, the policy configuration uses a `validation-database` match condition. This match condition triggers a query in the RV database for the validation state of a prefix in a given routing instance. The default operation is to query the validation database matching the routing instance. If no route validation instance is found, the primary instance is queried.

In the following BGP import policy, the `from validation-database` condition triggers a lookup in the router's RV database. An action is taken if the validation state is valid. The action is to accept the route and set the `validation-state` in the routing table to valid.

```
[edit protocols bgp]
import validation;
```

```
[edit policy-options]
policy-statement validation-1 {
    term valid {
        from {
            protocol bgp;
            validation-database valid; # Triggers a lookup in the RV database
        }
        then {
            validation-state valid; # Sets the validation state in the routing table
            accept;
        }
    }
}
```

Community Attribute to Announce RPKI Validation State to IBGP Neighbors

Prefix validation is done only for external BGP (EBGP) updates. Within an AS, you likely do not want to have an RPKI session running on every internal BGP (IBGP) router. Instead, you need a way to carry the validation state across the IBGP mesh so that all IBGP speakers have consistent information. This is accomplished by carrying the validation state in a non-transitive extended community. The community attribute announces and receives the validation state of a prefix between IBGP neighbors.

Junos OS supports the following well-known extended communities for route validation:

- origin-validation-state-valid
- origin-validation-state-invalid
- origin-validation-state-unknown

The following sample BGP import policy is configured on the router that has a session with an RPKI server.

Router With RPKI Session

```
policy-statement validation-1 {
    term valid {
        from {
            protocol bgp;
            validation-database valid;
        }
        then {
            validation-state valid;
            community add origin-validation-state-valid;
            accept;
        }
    }
}
```

The following sample BGP import policy is configured on an IBGP peer router that does not have a session with an RPKI server.

IBGP Peer Router Without RPKI Session

```
policy-statement validation-2 {
    term valid {
        from community origin-validation-state-valid;
        then validation-state valid;
    }
}
```

Nonstop Active Routing and Origin Validation

When you configure origin validation on a router that has dual Routing Engines and *nonstop active routing* is enabled, both the primary and the standby Routing Engines have a copy of the RV database. These two RV databases remain synchronized with each other.

The router does not maintain two identical sessions with the RPKI server. The RPKI-RTR protocol runs on the primary Routing Engine only. On the standby Routing Engine, the RPKI cache server session is always down.

The RV database is actively maintained by the primary Routing Engine through its session with the RPKI server. This database is replicated on the standby Routing Engine. Though the session is down on the standby Routing Engine , the replicated RV database does contain RV records. When the standby

Routing Engine switches over and becomes the primary Routing Engine, it already has a fully populated RV database.

To view the contents of the two databases, use the `show validation database` and `show validation replication database` commands.

Marking a Prefix Range as Never Allowed

The route validation model has one major shortcoming: It only provides positive updates. It can declare which AS is the legitimate owner of a prefix. However, it cannot explicitly convey a negative update, as in: This prefix is never originated by a given AS. This functionality can be provided to some extent using an AS 0 workaround.

The Junos OS implementation does not attempt to restrict its inputs from the cache. For example, an RV record with origin AS 0 is installed and matched upon just like any other. This enables a workaround to mark a prefix range as never allowed to be announced because AS 0 is not a valid AS. The AS in the RV record never matches the AS received from the EBGP peer. Thus, any matching prefix is marked invalid.

SEE ALSO

Example: Configuring Origin Validation for BGP | **1283**

Use Case and Benefit of Origin Validation for BGP

If an administrator of an autonomous system (AS) begins advertising all or part of another company's assigned network, BGP has no built-in method to recognize the error and respond in a way that would avoid service interruptions.

Suppose, for example, that an administrator in a customer network mistakenly advertises a route (let's say 10.65.153.0/24) directing traffic to the customer's service provider AS 1. This /24 route is a more specific route than the one used by the actual content provider (10.65.152.0/22) which directs traffic to AS 2. Because of the way routers work, most routers select the more specific route and send traffic to AS 1 instead of AS 2.

The hijacked prefix is seen widely across the Internet as transit routers propagate the updated path information. The invalid routes can be distributed broadly across the Internet as the routers in the default free zone (DFZ) carry the hijacked route. Eventually the correct AS path is restored to BGP peers, but in the meantime service interruptions are to be expected.

Because BGP relies on a transitive trust model, validation between customer and provider is important. In the example above, the service provider AS 1 did not validate the faulty advertisement for

10.65.153.0/24. By accepting this advertisement and readvertising it to its peers and providers, AS 1 was propagating the wrong route. The routers that received this route from AS 1 selected it because it was a more specific route. The actual content provider was advertising 10.65.152.0/22 before the mistake occurred. The /24 was a smaller (and more specific) advertisement. According to the usual BGP route selection process, the /24 was then chosen, effectively completing the hijack.

Even with fast detection and reaction of the content provider and cooperation with other providers, service for their prefix can be interrupted for many minutes up to several hours. The exact duration of the outage depends on your vantage point on the Internet. When these sorts of events occur, there is renewed interest in solutions to this vulnerability. BGP is fundamental to provider relationships and will not be going away anytime soon. This example demonstrates a solution that uses origin validation. This solution relies on cryptographic extensions to BGP and a distributed client-server model that avoids overtaxing router CPUs.

Origin validation helps to overcome the vulnerability of transitive trust by enabling a provider to limit the advertisements it accepts from a customer. The mechanics involve the communication of routing policies based on an extended BGP community attribute.

SEE ALSO

Understanding Origin Validation for BGP | **1274**

Example: Configuring Origin Validation for BGP

IN THIS SECTION

- Requirements | **1284**
- Overview | **1284**
- Configuration | **1286**
- Verification | **1299**

This example shows how to configure origin validation between BGP peers by ensuring that received route advertisements are sent (originated) from the expected autonomous system (AS). If the origin AS is validated, a policy can specify that the prefixes are, in turn, advertised.

Requirements

This example has the following hardware and software requirements:

- Resource public key infrastructure (RPKI) cache server, using third-party software to authenticate BGP prefixes.
- Junos OS Release 12.2 or later running on the routing device that communicates with the cache server over a TCP connection.

Overview

Sometimes routes are unintentionally advertised due to operator error. To prevent this security issue, you can configure BGP to validate the originating AS and reject these invalid announcements. This feature uses a cache server to authenticate prefixes or prefix ranges.

The following configuration statements enable origin AS validation:

```
[edit routing-options]
validation {
    group group-name {
        max-sessions number;
        session address {
            hold-time seconds;
            local-address local-ip-address;
            port port-number;
            preference number;
            record-lifetime seconds;
            refresh-time seconds;
            traceoptions {
            file filename <files number>     <size size> <(world-readable | no-world-readable)>;
                flag flag {
                    disable;
                    flag-modifier;
                }
        }
    }
    static {
        record destination {
            maximum-length prefix-length {
                origin-autonomous-system as-number {
                    validation-state (invalid | valid);
                }
            }
```

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

This example uses default settings for the validation parameters.

Most of the available configuration statements are optional. The required settings are as follows:

```
validation {
    group group-name {
        session address {
        }
    }
}
```

The `[edit routing-options validation static]` hierarchy level enables you to configure static records on a routing device, thus overwriting records received from an RPKI cache server.

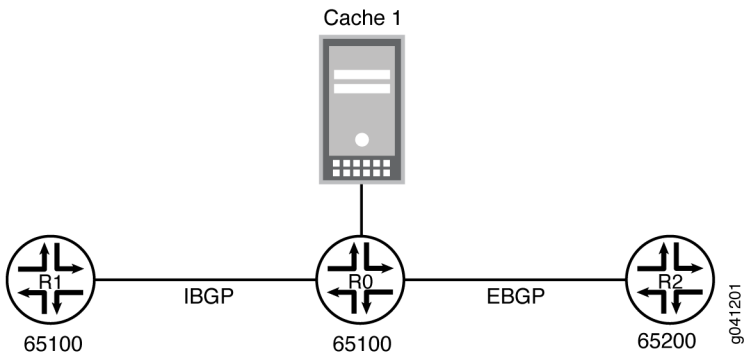
For example:

```
[edit routing-options validation]
user@R0# set static record 10.0.0.0/16 maximum-length 24 origin-autonomous-system 200 validation-
state valid
```

You can configure a routing policy that operates based on the validation state of a route prefix. You can use a community attribute to announce and receive the validation state of a prefix between external BGP (EBGP) and internal BGP (IBGP) peers. Using a routing policy might be more convenient on some routers than configuring a session with an RPKI server. This example demonstrates the use of the validation-state community attribute between IBGP peers.

Figure 90 on page 1286 shows the sample topology.

Figure 90: Topology for Origin Validation



In this example, Device R0 has an IBGP connection to Device R1 and an EBGP connection to Device R2. Device R0 receives route validation (RV) records from the cache server using the protocol defined in Internet draft draft-ietf-sidr-rpki-rtr-19, *The RPKI/Router Protocol* to send the RV records. The RPKI-Router Protocol runs over TCP. The RV records are used by Device R0 to build a local RV database. On Device R1, the validation state is set based on the BGP community called validation-state, which is received with the route.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1286**
- Configuring Device R0 | **1289**
- Configuring Device R1 | **1294**
- Configuring Device R2 | **1297**

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 R0

```
set interfaces ge-1/2/0 unit 2 description to-R1
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
```

```
set interfaces ge-1/2/1 unit 0 description to-R2
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/2 unit 0 description to-cache
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 10.0.1.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 10.0.1.1
set protocols bgp group int export send-direct
set protocols bgp group int neighbor 10.1.1.1
set protocols bgp group ext type external
set protocols bgp group ext import validation
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 65200
set protocols bgp group ext neighbor 10.0.0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/0.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set policy-options policy-statement validation term valid from protocol bgp
set policy-options policy-statement validation term valid from validation-database valid
set policy-options policy-statement validation term valid then validation-state valid
set policy-options policy-statement validation term valid then community add origin-validation-
state-valid
set policy-options policy-statement validation term valid then accept
set policy-options policy-statement validation term invalid from protocol bgp
set policy-options policy-statement validation term invalid from validation-database invalid
set policy-options policy-statement validation term invalid then validation-state invalid
set policy-options policy-statement validation term invalid then community add origin-validation-
state-invalid
set policy-options policy-statement validation term invalid then reject
set policy-options policy-statement validation term unknown from protocol bgp
set policy-options policy-statement validation term unknown then validation-state unknown
set policy-options policy-statement validation term unknown then community add origin-validation-
state-unknown
set policy-options policy-statement validation term unknown then accept
set policy-options community origin-validation-state-invalid members 0x4300:0.0.0.0:2
set policy-options community origin-validation-state-unknown members 0x4300:0.0.0.0:1
set policy-options community origin-validation-state-valid members 0x4300:0.0.0.0:0
set routing-options autonomous-system 65100
set routing-options validation group test session 10.0.0.10
```

Device R1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 10.1.1.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 10.1.1.1
set protocols bgp group int import validation-ibgp
set protocols bgp group int neighbor 10.0.1.1
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement validation-ibgp term valid from community origin-validation-
state-valid
set policy-options policy-statement validation-ibgp term valid then validation-state valid
set policy-options policy-statement validation-ibgp term invalid from community origin-
validation-state-invalid
set policy-options policy-statement validation-ibgp term invalid then validation-state invalid
set policy-options policy-statement validation-ibgp term unknown from community origin-
validation-state-unknown
set policy-options policy-statement validation-ibgp term unknown then validation-state unknown
set policy-options community origin-validation-state-invalid members 0x4300:0.0.0.0:2
set policy-options community origin-validation-state-unknown members 0x4300:0.0.0.0:1
set policy-options community origin-validation-state-valid members 0x4300:0.0.0.0:0
set routing-options autonomous-system 65100
```

Device R2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces lo0 unit 0 family inet address 192.168.2.3/32

set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 65100
set protocols bgp group ext neighbor 10.0.0.5
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct from protocol local
set policy-options policy-statement send-direct then accept
set routing-options autonomous-system 65200
```

Configuring Device R0

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 Junos OS CLI User Guide.

To configure Device R0:

1. Configure the interfaces.

```
[edit interfaces]
user@R0# set ge-1/2/0 unit 0 description to-R1
user@R0# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
user@R0# set ge-1/2/1 unit 0 description to-R2
user@R0# set ge-1/2/1 unit 0 family inet address 10.0.0.5/30
user@R0# set ge-1/2/2 unit 0 description to-cache
user@R0# set ge-1/2/2 unit 0 family inet address 10.0.0.9/30
user@R0# set lo0 unit 0 family inet address 10.0.1.1/32
```

2. Configure BGP.

   Apply the `send-direct` export policy so that direct routes are exported from the routing table into BGP.

   Apply the `validation` import policy to set the validation-state and BGP community attributes for all the routes imported (or received) from Device R0's EBGP peers.

   Configure an IBGP session with Device R1. Configure an EBGP session with Device R2.

```
[edit protocols bgp]
user@R0# set group int type internal
user@R0# set group int local-address 10.0.1.1
user@R0# set group int export send-direct
user@R0# set group int neighbor 10.1.1.1
user@R0# set group ext type external
user@R0# set group ext import validation
user@R0# set group ext export send-direct
user@R0# set group ext peer-as 65200
user@R0# set group ext neighbor 10.0.0.6
```

3. Configure OSPF (or another interior gateway protocol [IGP]) on the interface that faces the IBGP peer and on the loopback interface.

> **NOTE**: If you use the loopback interface address in the IBGP `neighbor` statement, you must enable an IGP on the loopback interface. Otherwise, the IBGP session is not established.

```
[edit protocols ospf area 0.0.0.0]
user@R0# set interface ge-1/2/0.0
user@R0# set interface lo0.0 passive
```

4. Configure the routing policy that exports direct routes from the routing table into BGP.

```
[edit policy-options policy-statement send-direct]
user@R0# set from protocol direct
user@R0# set then accept
```

5. Configure the routing policy that specifies attributes to be modified based on the validation state of each BGP route.

```
[edit policy-options policy-statement validation]
user@R0# set term valid from protocol bgp
user@R0# set term valid from validation-database valid
user@R0# set term valid then validation-state valid
user@R0# set term valid then community add origin-validation-state-valid
user@R0# set term valid then accept
user@R0# set term invalid from protocol bgp
user@R0# set term invalid from validation-database invalid
user@R0# set term invalid then validation-state invalid
user@R0# set term invalid then community add origin-validation-state-invalid
user@R0# set term invalid then reject
user@R0# set term unknown from protocol bgp
user@R0# set term unknown then validation-state unknown
user@R0# set term unknown then community add origin-validation-state-unknown
user@R0# set term unknown then accept
[edit policy-options]
user@R0# set community origin-validation-state-invalid members 0x4300:0.0.0.0:2
```

```
user@R0# set community origin-validation-state-unknown members 0x4300:0.0.0.0:1
user@R0# set community origin-validation-state-valid members 0x4300:0.0.0.0:0
```

6. Configure the session with the RPKI cache server.

```
[edit routing-options validation]
user@R0# set group test session 10.0.0.10
```

7. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R0# set autonomous-system 65100
```

Results

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

```
user@R0# show interfaces
ge-1/2/0 {
    unit 0 {
        description to-R1;
        family inet {
            address 10.0.0.2/30;
        }
    }
}
ge-1/2/1 {
    unit 0 {
        description to-R2;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
ge-1/2/2 {
    unit 0 {
        description to-cache;
```

```
        family inet {
            address 10.0.0.9/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.0.1.1/32;
        }
    }
}
```

```
user@R0# show protocols
bgp {
    group int {
        type internal;
        local-address 10.0.1.1;
        export send-direct;
        neighbor 10.1.1.1;
    }
    group ext {
        type external;
        import validation;
        export send-direct;
        peer-as 65200;
        neighbor 10.0.0.6;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/2/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R0# show policy-options
policy-statement send-direct {
```

```
    from protocol direct;
    then accept;
}
    policy-statement validation {
        term valid {
            from {
                protocol bgp;
                validation-database valid;
            }
            then {
                validation-state valid;
                community add origin-validation-state-valid;
                accept;
            }
        }
        term invalid {
            from {
                protocol bgp;
                validation-database invalid;
            }
            then {
                validation-state invalid;
                community add origin-validation-state-invalid;
                reject;
            }
        }
        term unknown {
            from protocol bgp;
            then {
                validation-state unknown;
                community add origin-validation-state-unknown;
                accept;
            }
        }
    }
community origin-validation-state-invalid members 0x4300:0.0.0.0:2;
community origin-validation-state-unknown members 0x4300:0.0.0.0:1;
```

```
community origin-validation-state-valid members 0x4300:0.0.0.0:0;
}
```

```
user@R0# show routing-options
autonomous-system 65100;
validation {
    group test {
        session 10.0.0.10;
    }
}
```

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

Configuring Device R1

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 Junos OS CLI User Guide.

To configure Device R1:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@R1# set ge-1/2/0 unit 0 family inet address 10.0.0.1/30
   user@R1# set lo0 unit 0 family inet address 10.1.1.1/32
   ```

2. Configure BGP.

   Apply the `validation-ibgp` import policy to set the validation-state and BGP community attributes for all the routes received from Device R1's IBGP peers.

   Configure an IBGP session with Device R0.

   ```
   [edit protocols bgp group int]
   user@R1# set type internal
   user@R1# set local-address 10.1.1.1
   ```

```
user@R1# set import validation-ibgp
user@R1# set neighbor 10.0.1.1
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface ge-1/2/0.0
user@R1# set interface lo0.0 passive
```

4. Configure the routing policy that specifies attributes to be modified based on the validation-state BGP community attribute of the BGP routes received from Device R0.

```
[edit policy-options policy-statement validation-ibgp]
user@R1# set term valid from community origin-validation-state-valid
user@R1# set term valid then validation-state valid
user@R1# set term invalid from community origin-validation-state-invalid
user@R1# set term invalid then validation-state invalid
user@R1# set term unknown from community origin-validation-state-unknown
user@R1# set term unknown then validation-state unknown
[edit policy-options]
user@R1# set community origin-validation-state-invalid members 0x4300:0.0.0.0:2
user@R1# set community origin-validation-state-unknown members 0x4300:0.0.0.0:1
user@R1# set community origin-validation-state-valid members 0x4300:0.0.0.0:0
```

5. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set autonomous-system 65100
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
ge-1/2/0 {
    unit 0 {
```

```
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.1.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group int {
        type internal;
        local-address 10.1.1.1;
        import validation-ibgp;
        neighbor 10.0.1.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/2/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement validation-ibgp {
    term valid {
        from community origin-validation-state-valid;
        then validation-state valid;
    }
    term invalid {
        from community origin-validation-state-invalid;
        then validation-state invalid;
```

```
    }
    term unknown {
        from community origin-validation-state-unknown;
        then validation-state unknown;
    }
}
community origin-validation-state-invalid members 0x4300:0.0.0.0:2;
community origin-validation-state-unknown members 0x4300:0.0.0.0:1;
community origin-validation-state-valid members 0x4300:0.0.0.0:0;
}
```

```
user@R1# show routing-options
autonomous-system 65100;
```

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

Configuring Device R2

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 Junos OS CLI User Guide.

To configure Device R2:

1. Configure the interfaces.

   Several addresses are configured on the loopback interface to serve as routes for demonstration purposes.

   ```
   [edit interfaces]
   user@R2# set ge-1/2/0 unit 0 family inet address 10.0.0.6/30
   user@R2# set lo0 unit 0 family inet address 172.16.1.1/32
   user@R2# set lo0 unit 0 family inet address 192.168.2.3/32
   ```

2. Configure BGP.

   ```
   [edit protocols bgp]
   user@R2# set group ext export send-direct
   ```

```
user@R2# set group ext peer-as 65100
user@R2# set group ext neighbor 10.0.0.5
```

3. Configure the routing policy.

```
[edit policy-options policy-statement send-direct]
user@R2# set from protocol direct
user@R2# set from protocol local
user@R2# set then accept
```

4. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 65200
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
ge-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.6/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.1.1/32;
            address 192.168.2.3/32;
        }
```

```
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        export send-direct;
        peer-as 65100;
        neighbor 10.0.0.5;
    }
}
```

```
user@R2# show policy-options
policy-statement send-direct {
    from protocol [ direct local ];
    then accept;
}
```

```
user@R2# show routing-options
autonomous-system 65200;
```

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

Verification

IN THIS SECTION

- Verifying That the Modified Attributes Are Displayed in the Routing Tables | **1300**
- Using Trace Operations | **1301**
- Displaying Validation Information | **1302**

Confirm that the configuration is working properly.

Verifying That the Modified Attributes Are Displayed in the Routing Tables

Purpose

Verify that the BGP routes on Device R0 and Device R1 have the expected validation states and the expected local preferences.

Action

From operational mode, enter the `show route` command.

```
user@R0> show route
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.1.1/32         *[Direct/0] 04:53:39
                    >    via lo0.1
10.1.1.1/32         *[OSPF/10] 04:50:53, metric 1
                    > to 10.0.0.1 via lt-1/2/0.2

10.0.0.0/30        *[Direct/0] 04:51:44
                    >    via lt-1/2/0.2
10.0.0.2/32        *[Local/0] 04:51:45
                         Local via lt-1/2/0.2
10.0.0.4/30        *[Direct/0] 04:51:44
                    >    via lt-1/2/0.5
                    [BGP/170] 02:24:57, localpref 100
                      AS path: 65200 I, validation-state: valid
                    > to 10.0.0.6 via lt-1/2/0.5
10.0.0.5/32        *[Local/0] 04:51:45
                         Local via lt-1/2/0.5
10.0.0.8/30        *[Direct/0] 03:01:28
                    >    via lt-1/2/0.9
10.0.0.9/32        *[Local/0] 04:51:45
                         Local via lt-1/2/0.9
172.16.1.1/32      *[BGP/170] 02:24:57, localpref 100
                      AS path: 65200 I, validation-state: invalid
                    > to 10.0.0.6 via lt-1/2/0.5
192.168.2.3/32     *[BGP/170] 02:24:57, localpref 100
                      AS path: 65200 I, validation-state: validation-state: unknown
                    > to 10.0.0.6 via lt-1/2/0.5
```

```
224.0.0.5/32       *[OSPF/10] 04:53:46, metric 1
                          MultiRecv
```

```
user@R1> show route
inet.0: 10 destinations, 12 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both


172.16.1.1/32      *[BGP/170] 00:40:52, localpref 100, from 1.0.1.1
                      AS path: 65200 I, validation-state: invalid
                    > to 10.0.0.2 via lt-1/2/0.1
192.168.2.3/32     *[BGP/170] 01:06:58, localpref 100, from 1.0.1.1
                      AS path: 65200 I, validation-state: unknown
                    > to 10.0.0.2 via lt-1/2/0.1
224.0.0.5/32       *[OSPF/10] 04:57:09, metric 1
                          MultiRecv
```

Meaning

The routes have the expected validation states and local-preference values, based on information received from the RPKI cache server.

Using Trace Operations

Purpose

Configure trace operations for origin validation, and monitor the results of a newly advertised route.

Action

- On Device R0, configure tracing.

```
[edit routing-options validation traceoptions]
user@R0# set file rv-tracing
user@R0# set flag all

user@R0# commit
```

- On Device R2, add a route by adding another address on the loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@R2# set address 10.4.4.4/32

user@R2# commit
```

- On Device R0, check the trace file.

```
user@R0> file show /var/log/rv-tracing
Jan 27 11:27:43.804803 rv_get_policy_state: rt 10.4.4.4/32 origin-as 65200, validation result
valid
Jan 27 11:27:43.944037 task_job_create_background: create prio 7 job Route-validation GC for
task Route Validation
Jan 27 11:27:43.986580 background dispatch running job Route-validation GC for task Route
Validation
Jan 27 11:27:43.987374 task_job_delete: delete background job Route-validation GC for task
Route Validation
Jan 27 11:27:43.987463 background dispatch completed job Route-validation GC for task Route
Validation
```

Meaning

Route validation is operating as expected.

Displaying Validation Information

Purpose

Run the various validation commands.

Action

```
user@R0> show validation statistics
Total RV records: 2
Total Replication RV records: 2
  Prefix entries: 2
  Origin-AS entries: 2
Memory utilization: 9789 bytes
```

```
Policy origin-validation requests: 114
  Valid: 32
  Invalid: 54
  Unknown: 28
BGP import policy reevaluation notifications: 156
  inet.0, 156
  inet6.0, 0
```

```
user@R0> show validation database
RV database for instance master

Prefix                Origin-AS Session                                   State   Mismatch
10.0.0.0/8-32               65200 10.0.0.10                               valid
172.0.0.0/8-12              65200 10.0.0.10                               invalid

  IPv4 records: 2
  IPv6 records: 0
```

```
user@R0> show validation replication database
RRV replication database for instance master

Prefix                Origin-AS Session                                   State
10.0.0.0/8-32               65200 10.0.0.10                               valid
172.0.0.0/8-12              65200 10.0.0.10                               invalid

  IPv4 records: 2
  IPv6 records: 0
```

```
user@R0> show validation group
master
```

```
  Group: test, Maximum sessions: 2
    Session 10.0.0.10, State: Connect, Preference: 100
```

```
user@R0> show validation session
Session                                 State   Flaps     Uptime #IPv4/IPv6 records
10.0.0.10                               Up          0   00:02:28 1/0
```

```
user@R0> request validation policy
  Enqueued 2 IPv4 records
  Enqueued 0 IPv6 records
```

SEE ALSO

Understanding Origin Validation for BGP | 1274

CHAPTER 11

Configuring BGP Route Flap Damping and Error Handling

IN THIS CHAPTER

- BGP Session and Route Flaps | **1306**
- BGP Error Messages | **1370**
- BFD for BGP Sessions | **1387**

BGP Session and Route Flaps

IN THIS SECTION

- Understanding BGP Session Resets | **1306**
- Example: Preventing BGP Session Flaps When VPN Families Are Configured | **1307**
- Understanding Damping Parameters | **1316**
- Example: Configuring BGP Route Flap Damping Parameters | **1318**
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family | **1332**
- Understanding BGP-Static Routes for Preventing Route Flaps | **1347**
- Configuring BGP-Static Routes for Preventing Route Flaps | **1348**
- Example: Configuring BGP-Static Routes to Prevent Route Flaps | **1349**

Understanding BGP Session Resets

Certain configuration actions and events cause BGP sessions to be reset (dropped and then reestablished).

If you configure both route reflection and VPNs on the same routing device, the following modifications to the route reflection configuration cause current BGP sessions to be reset:

- Adding a cluster ID—If a BGP session shares the same autonomous system (AS) number with the group where you add the cluster ID, all BGP sessions are reset regardless of whether the BGP sessions are contained in the same group.
- Creating a new route reflector—If you have an internal BGP (IBGP) group with an AS number and create a new route reflector group with the same AS number, all BGP sessions in the IBGP group and the new route reflector group are reset.
- Changing configuration statements that affect BGP peers, such as renaming a BGP group, resets the BGP sessions.
- If you change the address family specified in the `[edit protocols bgp family]` hierarchy level, all current BGP sessions on the routing device are dropped and then reestablished.

Example: Preventing BGP Session Flaps When VPN Families Are Configured

IN THIS SECTION

- Requirements | 1307
- Overview | 1307
- Configuration | 1310
- Verification | 1315

This example shows a workaround for a known issue in which BGP sessions sometimes go down and then come back up (in other words, flap) when virtual private network (VPN) families are configured. If any VPN family (for example, `inet-vpn`, `inet6-vpn`, `inet-mpvn`, `inet-mdt`, `inet6-mpvn`, `l2vpn`, `iso-vpn`, and so on) is configured on a BGP master instance, a flap of either a route reflector (RR) internal BGP (IBGP) session or an external BGP (EBGP) session causes flaps of other BGP sessions configured with the same VPN family.

Requirements

Before you begin:

- Configure router interfaces.
- Configure an interior gateway protocol (IGP).
- Configure BGP.
- Configure VPNs.

Overview

IN THIS SECTION

- Topology | 1309

When a router or switch is configured as either a route reflector (RR) or an AS boundary router (an external BGP peer) and a VPN family (for example, the `family inet-vpn unicast` statement) is configured, a flap of either the RR IBGP session or the EBGP session causes flaps of all other BGP sessions that are configured with the `family inet-vpn unicast` statement. This example shows how to prevent these unnecessary session flaps.

The reason for the flapping behavior is related to BGP operation in Junos OS when originating VPN routes.

BGP has the following two modes of operation with respect to originating VPN routes:

- If BGP does not need to propagate VPN routes because the session has no EBGP peer and no RR clients, BGP exports VPN routes directly from the *instance*.inet.0 routing table to other PE routers. This behavior is efficient in that it avoids the creation of two copies of many routes (one in the *instance*.inet.0 table and one in the bgp.l3vpn.0 table).
- If BGP does need to propagate VPN routes because the session has an EBGP peer or RR clients, BGP first exports the VPN routes from the *instance*.inet.0 table to the bgp.l3vpn.0 table. Then BGP exports the routes to other PE routers. In this scenario, two copies of the route are needed to enable best-route selection. A PE router might receive the same VPN route from a CE device and also from an RR client or EBGP peer.

NOTE: The route export is not performed if the route in *instance*.inet.0 is a secondary route. In Junos OS, a route is only exported one time from one routing table as a primary route to another routing table as a secondary route. Because the route in *instance*.inet.0 is already a secondary route, it is not allowed to be moved again to the bgp.l3vpn.0 table, as needed to be advertised. The route does not reach the bgp.l3vpn.0 table and thus is not advertised. One workaround is to send the routes that should be advertised to inet.0 so that they are advertised.

When, because of a configuration change, BGP transitions from needing two copies of a route to not needing two copies of a route (or the reverse), all sessions over which VPN routes are exchanged go down and then come back up. Although this example focuses on the `family inet-vpn unicast` statement, the concept applies to all VPN network layer reachability information (NLRI) families. This issue impacts logical systems as well. All BGP sessions in the master instance related to the VPN NLRI family are brought down to implement the table advertisement change for the VPN NLRI family. Changing an RR to a non-RR or the reverse (by adding or removing the `cluster` statement) causes the table advertisement change. Also, configuring the first EBGP session or removing the EBGP session from the configuration in the master instance for a VPN NLRI family causes the table advertisement change.

The way to prevent these unnecessary session flaps is to configure an extra RR client or EBGP session as a passive session with a neighbor address that does not exist. This example focuses on the EBGP case, but the same workaround works for the RR case.

When a session is passive, the routing device does not send Open requests to a peer. Once you configure the routing device to be passive, the routing device does not originate the TCP connection. However, when the routing device receives a connection from the peer and an Open message, it replies with another BGP Open message. Each routing device declares its own capabilities.

Topology

Figure 91 on page 1309 shows the topology for the EBGP case. Router R1 has an IBGP session with Routers R2 and R3 and an EBGP session with Router R4. All sessions have the `family inet-vpn unicast` statement configured. If the R1-R4 EBGP session flaps, the R1-R2 and R1-R3 BGP sessions flap also.

Figure 91: Topology for the EBGP Case

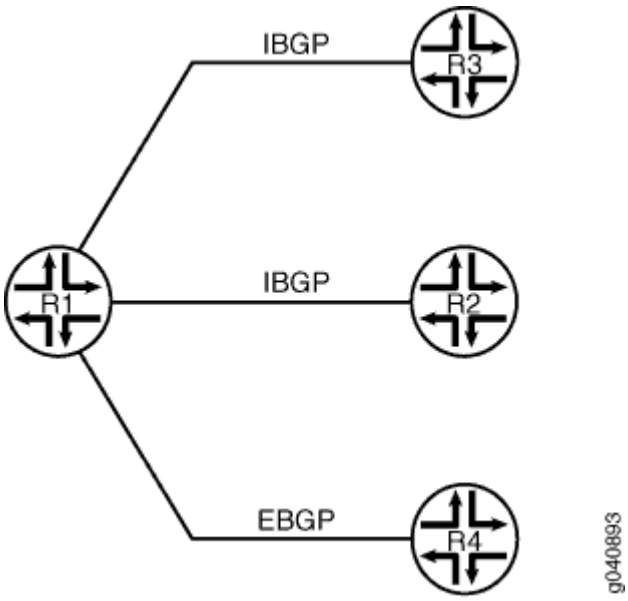
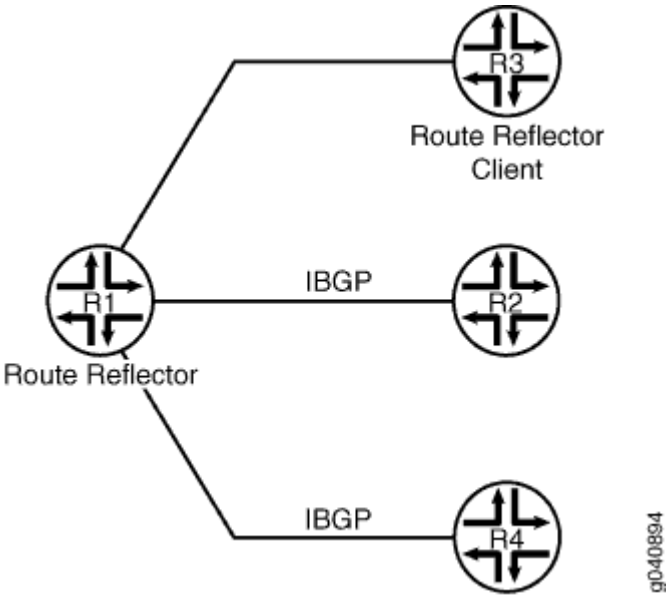


Figure 92 on page 1310 shows the topology for the RR case. Router R1 is the RR, and Router R3 is the client. Router R1 has IBGP sessions with Routers R2 and R3. All sessions have the `family inet-vpn unicast` statement configured. If the R1-R3 session flaps, the R1-R2 and R1-R4 sessions flap also.

Figure 92: Topology for the RR Case



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1310
- Procedure | 1311
- Procedure | 1312
- Procedure | 1314

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.

```
set protocols bgp family inet-vpn unicast
set protocols bgp family l2vpn signaling
set protocols bgp group R1-R4 type external
set protocols bgp group R1-R4 local-address 4.4.4.2
set protocols bgp group R1-R4 neighbor 4.4.4.1 peer-as 200
```

```
set protocols bgp group R1-R2-R3 type internal
set protocols bgp group R1-R2-R3 log-updown
set protocols bgp group R1-R2-R3 local-address 15.15.15.15
set protocols bgp group R1-R2-R3 neighbor 12.12.12.12
set protocols bgp group R1-R2-R3 neighbor 13.13.13.13
set protocols bgp group Fake type external
set protocols bgp group Fake passive
set protocols bgp group Fake neighbor 100.100.100.100 peer-as 500
```

Procedure

Step-by-Step Procedure

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

To configure the EBGP scenario:

1. Configure one or more VPN families.

```
[edit protocols bgp]
user@R1# set family inet-vpn unicast
user@R1# set family l2vpn signaling
```

2. Configure the EBGP session.

```
[edit protocols bgp]
user@R1# set group R1-R4 type external
user@R1# set group R1-R4 local-address 4.4.4.2
user@R1# set group R1-R4 neighbor 4.4.4.1 peer-as 200
```

3. Configure the IBGP sessions.

```
[edit protocols bgp]
user@R1# set group R1-R2-R3 type internal
user@R1# set group R1-R2-R3 local-address 15.15.15.15
user@R1# set group R1-R2-R3 neighbor 12.12.12.12
user@R1# set group R1-R2-R3 neighbor 13.13.13.13
```

4. (Optional) Configure BGP so that it generates a `syslog` message whenever a BGP peer makes a state transition.

```
[edit protocols bgp]
user@R1# set group R1-R2-R3 log-updown
```

Enabling the `log-updown` statement causes BGP state transitions to be logged at `warning` level.

Procedure

Step-by-Step Procedure

To verify that unnecessary session flaps are occurring:

1. Run the `show bgp summary` command to verify that the sessions have been established.

```
user@R1> show bgp summary
Groups: 2 Peers: 3 Down peers: 0
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
inet.0      0         0         0          0       0          0
Peer        AS  InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Accepted/Damped...
4.4.4.1     200 6     5      0    0     1:08 Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
12.12.12.12 100 3     7      0    0     1:18 Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
13.13.13.13 100 3     6      0    0     1:14 Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
```

2. Deactivate the EBGP session.

```
user@R1# deactivate group R1-R4
user@R1# commit
```

```
Mar 10 18:27:40 R1: rpd[1464]: bgp_peer_delete:6589: NOTIFICATION sent to 4.4.4.1 (External
AS 200): code 6 (Cease) subcode 3 (Peer Unconfigured), Reason: Peer Deletion
Mar 10 18:27:40 R1: rpd[1464]: bgp_adv_main_update:7253: NOTIFICATION sent to 12.12.12.12
(Internal AS 100): code 6 (Cease) subcode 6 (Other Configuration Change), Reason:
Configuration change - VPN table advertise
Mar 10 18:27:40 R1: rpd[1464]: bgp_adv_main_update:7253: NOTIFICATION sent to 13.13.13.13
(Internal AS 100): code 6 (Cease) subcode 6 (Other Configuration Change), Reason:
Configuration change - VPN table advertise
```

3. Run the `show bgp summary` command to view the session flaps.

```
user@R1> show bgp summary
Groups: 1 Peers: 2 Down peers: 2
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
inet.0      0         0         0          0       0          0
Peer        AS   InPkt  OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Accepted/
Damped...
12.12.12.12 100  4      9      0    1     19   Active
13.13.13.13 100  4      8      0    1     19   Active
```

```
user@R1> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
inet.0      0         0         0          0       0          0
Peer        AS  InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Accepted/Damped...
12.12.12.12 100 2     3      0    1     0    Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
13.13.13.13 100 2     3      0    1     0    Establ
```

```
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
```

Procedure

Step-by-Step Procedure

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

To prevent unnecessary BGP session flaps:

1. Add a passive EBGP session with a neighbor address that does not exist in the peer autonomous system (AS).

```
[edit protocols bgp]
user@R1# set group Fake type external
user@R1# set group Fake passive
user@R1# set neighbor 100.100.100.100 peer-as 500
```

2. Run the `show bgp summary` command to verify that the real sessions have been established and the passive session is idle.

```
user@R1> show bgp summary
Groups: 3 Peers: 4 Down peers: 1
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
Peer            AS  InPkt OutPkt OutQ Flaps Last Up/Dwn   State|#Active/Received/Accepted/
Damped...
4.4.4.1         200 9500  9439   0    0     2d   23:14:23 Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
12.12.12.12     100 10309 10239  0    0     3d    5:17:49 Establ
bgp.l3vpn.0: 0/0/0/0
13.13.13.13     100 10306 10241  0    0     3d    5:18:25 Establ
bgp.l3vpn.0: 0/0/0/0
100.100.100.100 500 0     0      0    0     2d   23:38:52 Idle
```

Verification

IN THIS SECTION

- Bringing Down the EBGP Session | 1315
- Verifying That the IBGP Sessions Remain Up | 1315

Confirm that the configuration is working properly.

Bringing Down the EBGP Session

Purpose

Try to cause the flap issue after the workaround is configured.

Action

```
user@R1# deactivate group R1-R4
user@R1# commit
```

Verifying That the IBGP Sessions Remain Up

Purpose

Make sure that the IBGP sessions do not flap after the EBGP session is deactivated.

Action

```
user@R1> show bgp summary
Groups: 2 Peers: 3 Down peers: 1
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
Peer            AS  InPkt OutPkt OutQ Flaps Last Up/Dwn    State|#Active/Received/Accepted/
Damped...
12.12.12.12     100 10312 10242  0    0     3d    5:19:01  Establ
```

```
bgp.l3vpn.0: 0/0/0/0
13.13.13.13     100 10309 10244  0    0     3d    5:19:37  Establ
bgp.l3vpn.0: 0/0/0/0
100.100.100.100 500 0     0      0    0     2d   23:40:04  Idle
```

```
user@R1> show bgp summary
Groups: 3 Peers: 4 Down peers: 1
Table       Tot Paths Act Paths Suppressed History Damp State Pending
bgp.l3vpn.0 0         0         0          0       0          0
bgp.l2vpn.0 0         0         0          0       0          0
Peer            AS  InPkt OutPkt OutQ Flaps Last Up/Dwn    State|#Active/Received/Accepted/
Damped...
4.4.4.1         200 5     4      0    0     28             Establ
bgp.l3vpn.0: 0/0/0/0
bgp.l2vpn.0: 0/0/0/0
12.12.12.12     100 10314 10244  0    0     3d    5:19:55  Establ
bgp.l3vpn.0: 0/0/0/0
13.13.13.13     100 10311 10246  0    0     3d    5:20:31  Establ
bgp.l3vpn.0: 0/0/0/0
100.100.100.100 500 0     0      0    0     2d   23:40:58  Idle
```

SEE ALSO

> *Understanding Virtual Routing and Forwarding Tables*
>
> KB20870

Understanding Damping Parameters

BGP *route flapping* describes the situation in which BGP systems send an excessive number of update messages to advertise network reachability information. BGP *flap damping* is a method of reducing the number of update messages sent between BGP peers, thereby reducing the load on these peers, without adversely affecting the route convergence time for stable routes.

Flap damping reduces the number of update messages by marking routes as ineligible for selection as the active or preferable route. Marking routes in this way leads to some delay, or *suppression*, in the propagation of route information, but the result is increased network stability. You typically apply flap damping to external BGP (EBGP) routes (routes in different ASs). You can also apply flap damping within

a confederation, between confederation member ASs. Because routing consistency within an AS is important, do not apply flap damping to internal BGP (IBGP) routes. (If you do, it is ignored.)

There is an exception that rule. Starting in Junos OS Release 12.2, you can apply flap damping at the address family level. In a Junos OS Release 12.2 or later installation, when you apply flap damping at the address family level, it works for both IBGP and EBGP.

By default, route flap damping is not enabled. Damping is applied to external peers and to peers at confederation boundaries.

When you enable damping, default parameters are applied, as summarized in Table 11 on page 1317.

Table 11: Damping Parameters

Damping Parameter	Description	Default Value	Possible Values
half-life ***minutes***	Decay half-life—Number of minutes after which an arbitrary value is halved if a route stays stable.	**15** (minutes)	**1** through **45**
max-suppress ***minutes***	Maximum hold-down time for a route, in minutes.	**60** (minutes)	**1** through **720**
reuse	Reuse threshold—Arbitrary value below which a suppressed route can be used again.	**750**	**1** through **20,000**
suppress	Cutoff (suppression) threshold—Arbitrary value above which a route can no longer be used or included in advertisements.	**3000**	**1** through **20,000**

To change the default BGP flap damping values, you define actions by creating a named set of damping parameters and including it in a routing policy with the damping action. For the damping routing policy to work, you also must enable BGP route flap damping.

SEE ALSO

Understanding Routing Policies

Example: Configuring BGP Route Flap Damping Parameters

IN THIS SECTION

- Requirements | 1318
- Overview | 1318
- Configuration | 1319
- Verification | 1325

This example shows how to configure damping parameters.

Requirements

Before you begin, configure router interfaces and configure routing protocols.

Overview

This example has three routing devices. Device R2 has external BGP (EBGP) connections with Device R1 and Device R3.

Device R1 and Device R3 have some static routes configured for testing purposes, and these static routes are advertised through BGP to Device R2.

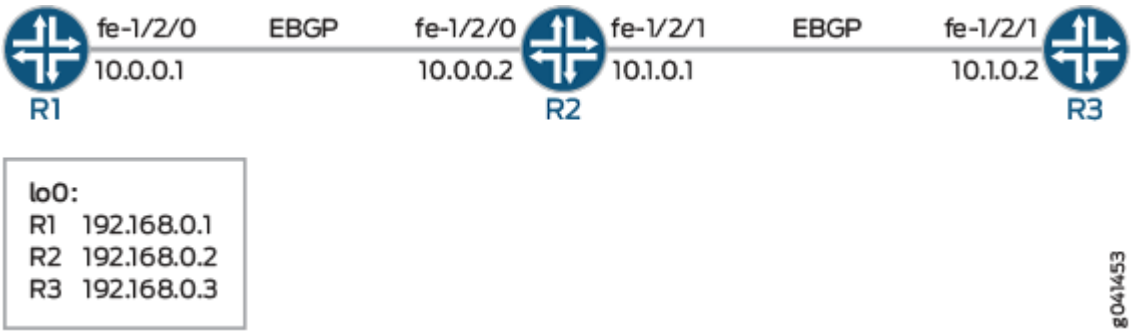
Device R2 damps routes received from Device R1 and Device R3 according to these criteria:

- Damp all prefixes with a mask length equal to or greater than 17 more aggressively than routes with a mask length between 9 and 16.
- Damp routes with a mask length between 0 and 8, inclusive, less than routes with a mask length greater than 8.
- Do not damp the 10.128.0.0/9 prefix at all.

The routing policy is evaluated when routes are being exported from the routing table into the forwarding table. Only the active routes are exported from the routing table.

Figure 93 on page 1319 shows the sample network.

Figure 93: BGP Flap Damping Topology



"CLI Quick Configuration" on page 1319 shows the configuration for all of the devices in Figure 93 on page 1319.

The section "No Link Title" on page 1321 describes the steps on Device R2.

Configuration

IN THIS SECTION

- Procedure | 1319

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-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 ext type external
set protocols bgp group ext export send-direct-and-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct-and-static term 1 from protocol direct
set policy-options policy-statement send-direct-and-static term 1 from protocol static
```

```
set policy-options policy-statement send-direct-and-static term 1 then accept
set routing-options static route 172.16.0.0/16 reject
set routing-options static route 172.16.128.0/17 reject
set routing-options static route 172.16.192.0/20 reject
set routing-options static route 10.0.0.0/9 reject
set routing-options static route 172.16.233.0/7 reject
set routing-options static route 10.224.0.0/11 reject
set routing-options static route 0.0.0.0/0 reject
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp damping
set protocols bgp group ext type external
set protocols bgp group ext import damp
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement damp term 1 from route-filter 10.128.0.0/9 exact damping dry
set policy-options policy-statement damp term 1 from route-filter 0.0.0.0/0 prefix-length-
range /0-/8 damping timid
set policy-options policy-statement damp term 1 from route-filter 0.0.0.0/0 prefix-length-
range /17-/32 damping aggressive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options damping aggressive half-life 30
set policy-options damping aggressive suppress 2500
set policy-options damping timid half-life 5
set policy-options damping dry disable
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct-and-static
set protocols bgp group ext peer-as 200
```

```
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct-and-static term 1 from protocol direct
set policy-options policy-statement send-direct-and-static term 1 from protocol static
set policy-options policy-statement send-direct-and-static term 1 then accept
set routing-options static route 10.128.0.0/9 reject
set routing-options autonomous-system 300
```

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 Junos OS CLI User Guide.

To configure damping parameters:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30
user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the BGP neighbors.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 100
user@R2# set neighbor 10.1.0.2 peer-as 300
```

3. Create and configure the damping parameter groups.

```
[edit policy-options]
user@R2# set damping aggressive half-life 30
user@R2# set damping aggressive suppress 2500
user@R2# set damping timid half-life 5
user@R2# set damping dry disable
```

4. Configure the damping policy.

```
[edit policy-options policy-statement damp term 1]
user@R2# set from route-filter 10.128.0.0/9 exact damping dry
user@R2# set from route-filter 0.0.0.0/0 prefix-length-range /0-/8 damping timid
user@R2# set from route-filter 0.0.0.0/0 prefix-length-range /17-/32 damping aggressive
```

5. Enable damping for BGP.

```
[edit protocols bgp]
user@R2# set damping
```

6. Apply the policy as an import policy for the BGP neighbor.

```
[edit protocols bgp group ext]
user@R2# set import damp
```

NOTE: You can refer to the same routing policy one or more times in the same or different `import` statements.

7. Configure an export policy.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

8. Apply the export policy.

```
[edit protocols bgp group ext]
user@R2# set export send-direct
```

9. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

Results

From configuration mode, confirm your configuration by issuing the `show interfaces`, `show protocols`, `show policy-options`, 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@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    damping;
    group ext {
        type external;
        import damp;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
        neighbor 10.1.0.2 {
            peer-as 300;
```

```
        }
    }
}
```

```
user@R2# show policy-options
policy-statement damp {
    term 1 {
        from {
            route-filter 10.128.0.0/9 exact damping dry;
            route-filter 0.0.0.0/0 prefix-length-range /0-/8 damping timid;
            route-filter 0.0.0.0/0 prefix-length-range /17-/32 damping aggressive;
        }
    }
}
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
damping aggressive {
    half-life 30;
    suppress 2500;
}
damping timid {
    half-life 5;
}
damping dry {
    disable;
}
```

```
user@R2# show routing-options
autonomous-system 200;
```

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

Verification

IN THIS SECTION

- Causing Some Routes to Flap | 1325
- Checking the Route Flaps | 1326
- Verifying Route Flap Damping | 1327
- Displaying the Details of a Damped Route | 1328
- Verifying That Default Damping Parameters Are in Effect | 1329
- Filtering the Damping Information | 1330

Confirm that the configuration is working properly.

Causing Some Routes to Flap

Purpose

To verify your route flap damping policy, some routes must flap. Having a live Internet feed almost guarantees that a certain number of route flaps will be present. If you have control over a remote system that is advertising the routes, you can modify the advertising router's policy to effect the advertisement and withdrawal of all routes or of a given prefix. In a test environment, you can cause routes to flap by clearing the BGP neighbors or by restarting the routing process on the BGP neighbors, as shown here.

Action

From operational mode on Device R1 and Device R3, enter the `restart routing` command.

CAUTION: Use this command cautiously in a production network.

```
user@R1> restart routing

R1 started, pid 10474
```

```
user@R3> restart routing

R3 started, pid 10478
```

Meaning

On Device R2, all of the routes from the neighbors are withdrawn and re-advertised.

Checking the Route Flaps

Purpose

View the number of neighbor flaps.

Action

From operational mode, enter the `show bgp summary` command.

```
user@R2> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                      12          1         11          0         11          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.0.0.1                100         10         10       0       4        2:50
0/9/0/9              0/0/0/0
10.1.0.2                300         10         10       0       4        2:53
1/3/1/2              0/0/0/0
```

Meaning

This output was captured after the routing process was restarted on Device R2's neighbors four times.

Verifying Route Flap Damping

Purpose

Verify that routes are being hidden due to damping.

Action

From operational mode, enter the `show route damping suppressed` command.

```
user@R2> show route damping suppressed

inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0           [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
10.0.0.0/9          [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
10.0.0.0/30         [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
10.1.0.0/30         [BGP ] 00:00:15, localpref 100
                      AS path: 300 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
10.224.0.0/11       [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.0.0/16       [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.128.0/17     [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.192.0/20     [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
```

```
                    > to 10.0.0.1 via fe-1/2/0.0
192.168.0.1/32      [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
192.168.0.3/32      [BGP ] 00:00:15, localpref 100
                      AS path: 300 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
172.16.233.0/7         [BGP ] 00:00:12, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
```

Meaning

The output shows some routing instability. Eleven routes are hidden due to damping.

Displaying the Details of a Damped Route

Purpose

Display the details of damped routes.

Action

From operational mode, enter the `show route damping suppressed 172.16.192.0/20 detail` command.

```
user@R2> show route damping suppressed 172.16.192.0/20 detail

inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
172.16.192.0/20 (1 entry, 0 announced)
         BGP                 /-101
                Next hop type: Router, Next hop index: 758
                Address: 0x9414484
                Next-hop reference count: 9
                Source: 10.0.0.1
                Next hop: 10.0.0.1 via fe-1/2/0.0, selected
                Session Id: 0x100201
                State: <Hidden Ext>
                Local AS:   200 Peer AS:   100
                Age: 52
                Validation State: unverified
                Task: BGP_100.10.0.0.1+55922
```

```
                AS path: 100 I
                Localpref: 100
                Router ID: 192.168.0.1
                Merit (last update/now): 4278/4196
                damping-parameters: aggressive
                Last update:       00:00:52 First update:       01:01:55
                Flaps: 8
                Suppressed. Reusable in:       01:14:40
                Preference will be: 170
```

Meaning

This output indicates that the displayed route has a mask length that is equal to or greater than /17, and confirms that it has been correctly mapped to the aggressive damping profile. You can also see the route's current (and last) figure of merit value, and when the route is expected to become active if it remains stable.

Verifying That Default Damping Parameters Are in Effect

Purpose

Locating a damped route with a /16 mask confirms that the default parameters are in effect.

Action

From operational mode, enter the `show route damping suppressed detail | match 0/16` command.

```
user@R2> show route damping suppressed detail | match 0/16

172.16.0.0/16 (1 entry, 0 announced)
```

```
user@R2> show route damping suppressed 172.16.0.0/16 detail

inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
172.16.0.0/16 (1 entry, 0 announced)
         BGP                /-101
                Next hop type: Router, Next hop index: 758
                Address: 0x9414484
                Next-hop reference count: 9
```

```
                Source: 10.0.0.1
                Next hop: 10.0.0.1 via fe-1/2/0.0, selected
                Session Id: 0x100201
                State: <Hidden Ext>
                Local AS:   200 Peer AS:   100
                Age: 1:58
                Validation State: unverified
                Task: BGP_100.10.0.0.1+55922
                AS path: 100 I
                Localpref: 100
                Router ID: 192.168.0.1
                Merit (last update/now): 3486/3202
                Default damping parameters used
                Last update:       00:01:58 First update:       01:03:01
                Flaps: 8
                Suppressed. Reusable in:       00:31:40
                Preference will be: 170
```

Meaning

Routes with a /16 mask are not impacted by the custom damping rules. Therefore, the default damping rules are in effect.

To repeat, the custom rules are as follows:

- Damp all prefixes with a mask length equal to or greater than 17 more aggressively than routes with a mask length between 9 and 16.
- Damp routes with a mask length between 0 and 8, inclusive, less than routes with a mask length greater than 8.
- Do not damp the 10.128.0.0/9 prefix at all.

Filtering the Damping Information

Purpose

Use OR groupings or cascaded piping to simplify the determination of what damping profile is being used for routes with a given mask length.

Action

From operational mode, enter the `show route damping suppressed` command.

```
user@R2> show route damping suppressed detail | match "0 announced | damp"

0.0.0.0/0 (1 entry, 0 announced)
                damping-parameters: timid
10.0.0.0/9 (1 entry, 0 announced)
                Default damping parameters used
                damping-parameters: aggressive
                damping-parameters: aggressive
10.224.0.0/11 (1 entry, 0 announced)
                Default damping parameters used
172.16.0.0/16 (1 entry, 0 announced)
                Default damping parameters used
172.16.128.0/17 (1 entry, 0 announced)
                damping-parameters: aggressive
172.16.192.0/20 (1 entry, 0 announced)
                damping-parameters: aggressive
192.168.0.1/32 (1 entry, 0 announced)
                damping-parameters: aggressive
192.168.0.3/32 (1 entry, 0 announced)
                damping-parameters: aggressive
172.16.233.0/7 (1 entry, 0 announced)
                damping-parameters: timid
```

Meaning

When you are satisfied that your EBGP routes are correctly associated with a damping profile, you can issue the `clear bgp damping` operational mode command to restore an active status to your damped routes, which will return your connectivity to normal operation.

SEE ALSO

Understanding Damping Parameters | 1316

Using Routing Policies to Damp BGP Route Flapping

Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family

IN THIS SECTION

- Requirements | 1332
- Overview | 1332
- Configuration | 1333
- Verification | 1345

This example shows how to configure an multiprotocol BGP multicast VPN (also called Next-Generation MVPN) with BGP route flap damping.

Requirements

This example uses Junos OS Release 12.2. BGP route flap damping support for MBGP MVPN, specifically, and on an address family basis, in general, is introduced in Junos OS Release 12.2.

Overview

IN THIS SECTION

- Topology | 1332

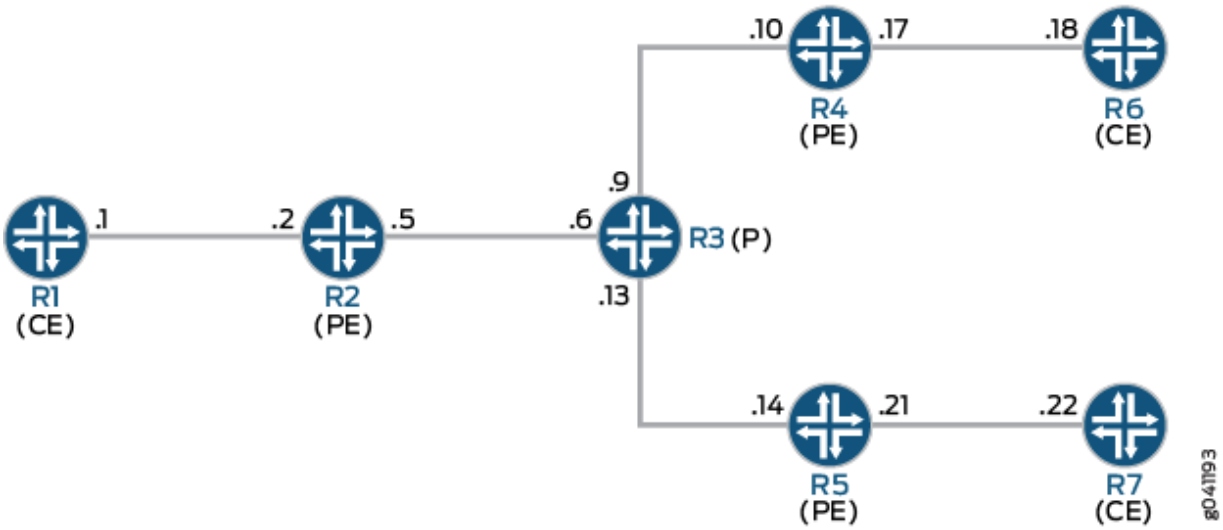
BGP route flap damping helps to diminish route instability caused by routes being repeatedly withdrawn and readvertised when a link is intermittently failing.

This example uses the default damping parameters and demonstrates an MBGP MVPN scenario with three provider edge (PE) routing devices, three customer edge (CE) routing devices, and one provider (P) routing device.

Topology

Figure 94 on page 1333 shows the topology used in this example.

Figure 94: MBGP MVPN with BGP Route Flap Damping



On PE Device R4, BGP route flap damping is configured for address family `inet-mvpn`. A routing policy called `dampPolicy` uses the `nlri-route-type` match condition to damp only MVPN route types 3, 4, and 5. All other MVPN route types are not damped.

This example shows the full configuration on all devices in the "CLI Quick Configuration" on page 1333 section. The "Configuring Device R4" on page 1338 section shows the step-by-step configuration for PE Device R4.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1333**
- Configuring Device R4 | **1338**
- Results | **1341**

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 R1

```
set interfaces ge-1/2/0 unit 1 family inet address 10.1.1.1/30
set interfaces ge-1/2/0 unit 1 family mpls
set interfaces lo0 unit 1 family inet address 172.16.1.1/32
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.1
set protocols pim rp static address 172.16.100.1
set protocols pim interface all
set routing-options router-id 172.16.1.1
```

Device R2

```
set interfaces ge-1/2/0 unit 2 family inet address 10.1.1.2/30
set interfaces ge-1/2/0 unit 2 family mpls
set interfaces ge-1/2/1 unit 5 family inet address 10.1.1.5/30
set interfaces ge-1/2/1 unit 5 family mpls
set interfaces vt-1/2/0 unit 2 family inet
set interfaces lo0 unit 2 family inet address 172.16.1.2/32
set interfaces lo0 unit 102 family inet address 172.16.100.1/32
set protocols mpls interface ge-1/2/1.5
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.2
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 172.16.1.4
set protocols bgp group ibgp neighbor 172.16.1.5
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/1.5
set protocols ldp interface ge-1/2/1.5
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface ge-1/2/0.2
set routing-instances vpn-1 interface vt-1/2/0.2
set routing-instances vpn-1 interface lo0.102
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 provider-tunnel ldp-p2mp
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
```

```
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.102 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/0.2
set routing-instances vpn-1 protocols pim rp static address 172.16.1.2 with 172.16.4.1100.1
set routing-instances vpn-1 protocols pim interface ge-1/2/0.2 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.2
set routing-options autonomous-system 1001
```

Device R3

```
set interfaces ge-1/2/0 unit 6 family inet address 10.1.1.6/30
set interfaces ge-1/2/0 unit 6 family mpls
set interfaces ge-1/2/1 unit 9 family inet address 10.1.1.9/30
set interfaces ge-1/2/1 unit 9 family mpls
set interfaces ge-1/2/2 unit 13 family inet address 10.1.1.13/30
set interfaces ge-1/2/2 unit 13 family mpls
set interfaces lo0 unit 3 family inet address 172.16.1.3/32
set protocols mpls interface ge-1/2/0.6
set protocols mpls interface ge-1/2/1.9
set protocols mpls interface ge-1/2/2.13
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/1.9
set protocols ospf area 0.0.0.0 interface ge-1/2/2.13
set protocols ldp interface ge-1/2/0.6
set protocols ldp interface ge-1/2/1.9
set protocols ldp interface ge-1/2/2.13
set protocols ldp p2mp
set routing-options router-id 172.16.1.3
```

Device R4

```
set interfaces ge-1/2/0 unit 10 family inet address 10.1.1.10/30
set interfaces ge-1/2/0 unit 10 family mpls
set interfaces ge-1/2/1 unit 17 family inet address 10.1.1.17/30
set interfaces ge-1/2/1 unit 17 family mpls
set interfaces vt-1/2/0 unit 4 family inet
set interfaces lo0 unit 4 family inet address 172.16.1.4/32
set interfaces lo0 unit 104 family inet address 172.16.100.1/32
set protocols rsvp interface all aggregate
set protocols mpls interface all
```

```
set protocols mpls interface ge-1/2/0.10
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.4
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling damping
set protocols bgp group ibgp neighbor 172.16.1.2 import dampPolicy
set protocols bgp group ibgp neighbor 172.16.1.5
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface lo0.4 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.10
set protocols ldp interface ge-1/2/0.10
set protocols ldp p2mp
set policy-options policy-statement dampPolicy term term1 from family inet-mvpn
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 3
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 4
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 5
set policy-options policy-statement dampPolicy term term1 then accept
set policy-options policy-statement dampPolicy then damping no-damp
set policy-options policy-statement dampPolicy then accept
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set policy-options damping no-damp disable
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.4
set routing-instances vpn-1 interface ge-1/2/1.17
set routing-instances vpn-1 interface lo0.104
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.104 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.17
set routing-instances vpn-1 protocols pim rp static address 172.16.100.1
set routing-instances vpn-1 protocols pim interface ge-1/2/1.17 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.4
set routing-options autonomous-system 64501
```

Device R5

```
set interfaces ge-1/2/0 unit 14 family inet address 10.1.1.14/30
set interfaces ge-1/2/0 unit 14 family mpls
set interfaces ge-1/2/1 unit 21 family inet address 10.1.1.21/30
set interfaces ge-1/2/1 unit 21 family mpls
set interfaces vt-1/2/0 unit 5 family inet
set interfaces lo0 unit 5 family inet address 172.16.1.5/32
set interfaces lo0 unit 105 family inet address 172.16.100.5/32
set protocols mpls interface ge-1/2/0.14
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.5
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 172.16.1.2
set protocols bgp group ibgp neighbor 172.16.1.4
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.14
set protocols ldp interface ge-1/2/0.14
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.5
set routing-instances vpn-1 interface ge-1/2/1.21
set routing-instances vpn-1 interface lo0.105
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.105 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.21
set routing-instances vpn-1 protocols pim rp static address 172.16.100.2
set routing-instances vpn-1 protocols pim interface ge-1/2/1.21 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.5
set routing-options autonomous-system 1001
```

Device R6

```
set interfaces ge-1/2/0 unit 18 family inet address 10.1.1.18/30
set interfaces ge-1/2/0 unit 18 family mpls
```

```
set interfaces lo0 unit 6 family inet address 172.16.1.6/32
set protocols sap listen 233.1.1.1
set protocols ospf area 0.0.0.0 interface lo0.6 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.18
set protocols pim rp static address 172.16.100.2
set protocols pim interface all
set routing-options router-id 172.16.1.6
```

Device R7

```
set interfaces ge-1/2/0 unit 22 family inet address 10.1.1.22/30
set interfaces ge-1/2/0 unit 22 family mpls
set interfaces lo0 unit 7 family inet address 172.16.1.7/32
set protocols ospf area 0.0.0.0 interface lo0.7 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.22
set protocols pim rp static address 172.16.100.2
set protocols pim interface all
set routing-options router-id 172.16.1.7
```

Configuring Device R4

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 Junos OS CLI User Guide.

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces]
user@R4# set ge-1/2/0 unit 10 family inet address 10.1.1.10/30
user@R4# set ge-1/2/0 unit 10 family mpls
user@R4# set ge-1/2/1 unit 17 family inet address 10.1.1.17/30
user@R4# set ge-1/2/1 unit 17 family mpls
user@R4# set vt-1/2/0 unit 4 family inet
user@R4# set lo0 unit 4 family inet address 172.16.1.4/32
user@R4# set lo0 unit 104 family inet address 172.16.100.4/32
```

2. Configure MPLS and the signaling protocols on the interfaces.

```
[edit protocols]
user@R4# set mpls interface all
user@R4# set mpls interface ge-1/2/0.10
user@R4# set rsvp interface all aggregate
user@R4# set ldp interface ge-1/2/0.10
user@R4# set ldp p2mp
```

3. Configure BGP.

   The BGP configuration enables BGP route flap damping for the `inet-mvpn` address family. The BGP configuration also imports into the routing table the routing policy called `dampPolicy`. This policy is applied to neighbor PE Device R2.

```
[edit protocols bgp group ibgp]
user@R4# set type internal
user@R4# set local-address 172.16.1.4
user@R4# set family inet-vpn unicast
user@R4# set family inet-vpn any
user@R4# set family inet-mvpn signaling damping
user@R4# set neighbor 172.16.1.2 import dampPolicy
user@R4# set neighbor 172.16.1.5
```

4. Configure an interior gateway protocol.

```
[edit protocols ospf]
user@R4# set traffic-engineering
[edit protocols ospf area 0.0.0.0]
user@R4# set interface all
user@R4# set interface lo0.4 passive
user@R4# set interface ge-1/2/0.10
```

5. Configure a damping policy that uses the `nlri-route-type` match condition to damp only MVPN route types 3, 4, and 5.

```
[edit policy-options policy-statement dampPolicy term term1]
user@R4# set from family inet-mvpn
user@R4# set from nlri-route-type 3
```

```
user@R4# set from nlri-route-type 4
user@R4# set from nlri-route-type 5
user@R4# set then accept
```

6. Configure the `damping` policy to disable BGP route flap damping.

   The `no-damp` policy (`damping no-damp disable`) causes any damping state that is present in the routing table to be deleted. The `then damping no-damp` statement applies the `no-damp` policy as an action and has no `from` match conditions. Therefore, all routes that are not matched by `term1` are matched by this term, with the result that all other MVPN route types are not damped.

```
[edit policy-options policy-statement dampPolicy]
user@R4# set then damping no-damp
user@R4# set then accept
[edit policy-options]
user@R4# set damping no-damp disable
```

7. Configure the `parent_vpn_routes` to accept all other BGP routes that are not from the `inet-mvpn` address family.

   This policy is applied as an OSPF export policy in the routing instance.

```
[edit policy-options policy-statement parent_vpn_routes]
user@R4# set from protocol bgp
user@R4# set then accept
```

8. Configure the VPN routing and forwarding (VRF) instance.

```
[edit routing-instances vpn-1]
user@R4# set instance-type vrf
user@R4# set interface vt-1/2/0.4
user@R4# set interface ge-1/2/1.17
user@R4# set interface lo0.104
user@R4# set route-distinguisher 100:100
user@R4# set vrf-target target:1:1
user@R4# set protocols ospf export parent_vpn_routes
user@R4# set protocols ospf area 0.0.0.0 interface lo0.104 passive
user@R4# set protocols ospf area 0.0.0.0 interface ge-1/2/1.17
user@R4# set protocols pim rp static address 172.16.100.2
```

```
user@R4# set protocols pim interface ge-1/2/1.17 mode sparse
user@R4# set protocols mvpn
```

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

```
[edit routing-options]
user@R4# set router-id 172.16.1.4
user@R4# set autonomous-system 1001
```

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

```
user@R4# commit
```

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@R4# show interfaces
ge-1/2/0 {
    unit 10 {
        family inet {
            address 10.1.1.10/30;
        }
        family mpls;
    }
}
ge-1/2/1 {
    unit 17 {
        family inet {
            address 10.1.1.17/30;
        }
        family mpls;
    }
}
vt-1/2/0 {
    unit 4 {
        family inet;
```

```
    }
}
lo0 {
    unit 4 {
        family inet {
            address 172.16.1.4/32;
        }
    }
    unit 104 {
        family inet {
            address 172.16.100.4/32;
        }
    }
}
```

```
user@R4# show protocols
rsvp {
    interface all {
        aggregate;
    }
}
mpls {
    interface all;
    interface ge-1/2/0.10;
}
bgp {
    group ibgp {
        type internal;
        local-address 172.16.1.4;
        family inet-vpn {
            unicast;
            any;
        }
        family inet-mvpn {
            signaling {
                damping;
            }
        }
        neighbor 172.16.1.2 {
            import dampPolicy;
        }
```

```
        neighbor 172.16.1.5;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface lo0.4 {
            passive;
        }
        interface ge-1/2/0.10;
    }
}
ldp {
    interface ge-1/2/0.10;
    p2mp;
}
```

```
user@R4# show policy-options
policy-statement dampPolicy {
    term term1 {
        from {
            family inet-mvpn;
            nlri-route-type [ 3 4 5 ];
        }
        then accept;
    }
    then {
        damping no-damp;
        accept;
    }
}
policy-statement parent_vpn_routes {
    from protocol bgp;
    then accept;
}
damping no-damp {
```

```
    disable;
}
```

```
user@R4# show routing-instances
vpn-1 {
    instance-type vrf;
    interface vt-1/2/0.4;
    interface ge-1/2/1.17;
    interface lo0.104;
    route-distinguisher 100:100;
    vrf-target target:1:1;
    protocols {
        ospf {
            export parent_vpn_routes;
            area 0.0.0.0 {
                interface lo0.104 {
                    passive;
                }
                interface ge-1/2/1.17;
            }
        }
        pim {
            rp {
                static {
                    address 172.16.100.2;
                }
            }
            interface ge-1/2/1.17 {
                mode sparse;
            }
        }
        mvpn;
    }
}
```

```
user@R4# show routing-optons
router-id 172.16.1.4;
autonomous-system 1001;
```

Verification

IN THIS SECTION

- Verifying That Route Flap Damping Is Disabled | 1345
- Verifying Route Flap Damping | 1346

Confirm that the configuration is working properly.

Verifying That Route Flap Damping Is Disabled

Purpose

Verify the presence of the `no-damp` policy, which disables damping for MVPN route types other than 3, 4, and 5.

Action

From operational mode, enter the `show policy damping` command.

```
user@R4> show policy damping
Default damping information:
  Halflife: 15 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 60 minutes
  Computed values:
    Merit ceiling: 12110
    Maximum decay: 6193
Damping information for "no-damp":
  Damping disabled
```

Meaning

The output shows that the default damping parameters are in effect and that the `no-damp` policy is also in effect for the specified route types.

Verifying Route Flap Damping

Purpose

Check whether BGP routes have been damped.

Action

From operational mode, enter the `show bgp summary` command.

```
user@R4> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0
                       6          6          0          0          0          0
bgp.l3vpn.2
                       0          0          0          0          0          0
bgp.mvpn.0
                       2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
172.16.1.2               1001       3159       3155       0       0    23:43:47 Establ
  bgp.l3vpn.0: 3/3/3/0
  bgp.l3vpn.2: 0/0/0/0
  bgp.mvpn.0: 1/1/1/0
  vpn-1.inet.0: 3/3/3/0
  vpn-1.mvpn.0: 1/1/1/0
172.16.1.5               1001       3157       3154       0       0    23:43:40 Establ
  bgp.l3vpn.0: 3/3/3/0
  bgp.l3vpn.2: 0/0/0/0
  bgp.mvpn.0: 1/1/1/0
  vpn-1.inet.0: 3/3/3/0
  vpn-1.mvpn.0: 1/1/1/0
```

Meaning

The Damp State field shows that zero routes in the bgp.mvpn.0 routing table have been damped. Further down, the last number in the State field shows that zero routes have been damped for BGP peer 172.16.1.2.

SEE ALSO

Understanding Damping Parameters | 1316

Using Routing Policies to Damp BGP Route Flapping

Example: Configuring BGP Route Flap Damping Parameters | 1318

Understanding BGP-Static Routes for Preventing Route Flaps

BGP-static routes can be configured to ensure that a prefix does not flap. BGP-static routes do not flap unless they are deleted manually. If the BGP-static routes are configured globally, then each neighbor, group, or all neighbors must be explicitly configured to receive them. Peer routers receive advertisements for these routes regardless of dynamic routing information learned by the advertising router for those prefixes. Despite being the active route, BGP-static routes are never advertised to a BGP neighbor for which they are not configured. You can specify any number of BGP-static routes in the configuration. You can also define a policy to specify which BGP-static routes need to be advertised and included in a BGP advertisement.

BGP-static routes are placed in the routing table. If the BGP-static routes are active routes (if there are no other routes for that prefix), they are placed in the forwarding table. These routes are advertised only to those BGP hosts that are configured to receive them. The configured BGP-static routes are not advertised by any other protocol besides BGP. Service providers who have one or more single-homed customers can configure BGP-static routes on a BGP network to advertise static paths for these customers.

NOTE: Configuring the advertisement of BGP-static routes at the neighbor level causes an internal group split. Configure the advertisement of BGP-static routes only at the global and group levels to keep the configuration simple. The configured BGP-static routes do not affect the VPN routes that are advertised.

If a BGP-static route is advertised to a neighbor, it is the only route advertised for the prefix. BGP-static routes are not considered as candidate routes for BGP multipath or protocol-independent multipath. They do not cause an aggregate or generated route to be added to the routing table.

CAUTION: Configuring BGP-static routes on networks that are accessible by multiple paths and are not the only point of access to all of the paths might cause traffic to be silently dropped or discarded. In a multihomed network, BGP-static routes can be configured on devices that are the only point of access to other paths. By default, all BGP-static routes that are advertised to the internal peers include a `local-pref` value of `0`

to mitigate the risk of a null route for multihomed networks. You can override this default value by setting an explicit `preference2` value on the BGP-static routes.

SEE ALSO

advertise-bgp-static

bgp-static

Configuring BGP-Static Routes for Preventing Route Flaps | 1348

Configuring BGP-Static Routes for Preventing Route Flaps

BGP-static routes are configured to ensure that routes to a customer network do not flap. The configured BGP-static routes are not advertised by any other protocol besides BGP. BGP-static routes are configured globally, but each neighbor, group, or all neighbors must be explicitly configured to receive them. Peer routers will receive advertisements for these routes regardless of dynamic routing information learned by the advertising router for those prefixes. You can specify any number of BGP-static routes in the configuration. You can also define a policy to specify which BGP-static routes need to be advertised.

Before you configure BGP-static routes:

1. Ensure that the IGP and BGP protocols are configured and working.
2. Ensure that BGP-static route that you configure is behind a customer router.

   Do not use BGP-static routes for prefixes that BGP uses to reach BGP neighbors.

To configure BGP-static routes:

1. Configure a BGP-static route for a customer router on a BGP network to advertise static paths for these customers.

   You can also configure other configuration options such as as-path, color, community, tag, and preference as needed.

   ```
   [edit routing-options]
   user@host# set bgp-static route destination-prefix
   ```

2. Configure the BGP groups or the BGP neighbors that are to receive the BGP-static route advertisements.

You can also configure this statement at a global level if you want every host on the BGP network to receive the BGP-static advertisements.

```
[edit protocols bgp]
user@host# set advertise-bgp-static
```

3. (Optional) Specify an additional export policy to control whether or not a given BGP-static route needs to be advertised.

   The policy is applied to the BGP-static route and not the active route.

```
[edit policy-options policy-statement policy name]
user@host# set from prefix-list xyz
user@host# set then accept
```

4. Apply the defined policy to a BGP group or neighbor.

```
[edit protocols bgp group group-name]
user@host# set advertise-bgp-static export policy name
```

SEE ALSO

advertise-bgp-static

bgp-static

Example: Configuring BGP-Static Routes to Prevent Route Flaps | 1349

Understanding BGP-Static Routes for Preventing Route Flaps | 1347

Example: Configuring BGP-Static Routes to Prevent Route Flaps

IN THIS SECTION

- Requirements | 1350
- Overview | 1350
- Configuration | 1352

- Verification | **1360**

This example shows how to configure BGP-static routes. BGP hosts advertise these BGP-static routes only to those neighbors who are configured to receive these routes. A BGP-static route is configured to ensure that a prefix does not flap. However, If the BGP-static routes are configured globally, then each neighbor, group, or all neighbors must be explicitly configured to receive them.

Requirements

This example uses the following hardware and software components:

- Seven MX Series routers with BGP enabled on the connected interfaces
- Junos OS Release 14.2 or later running on all devices

Overview

IN THIS SECTION

- Topology | **1351**

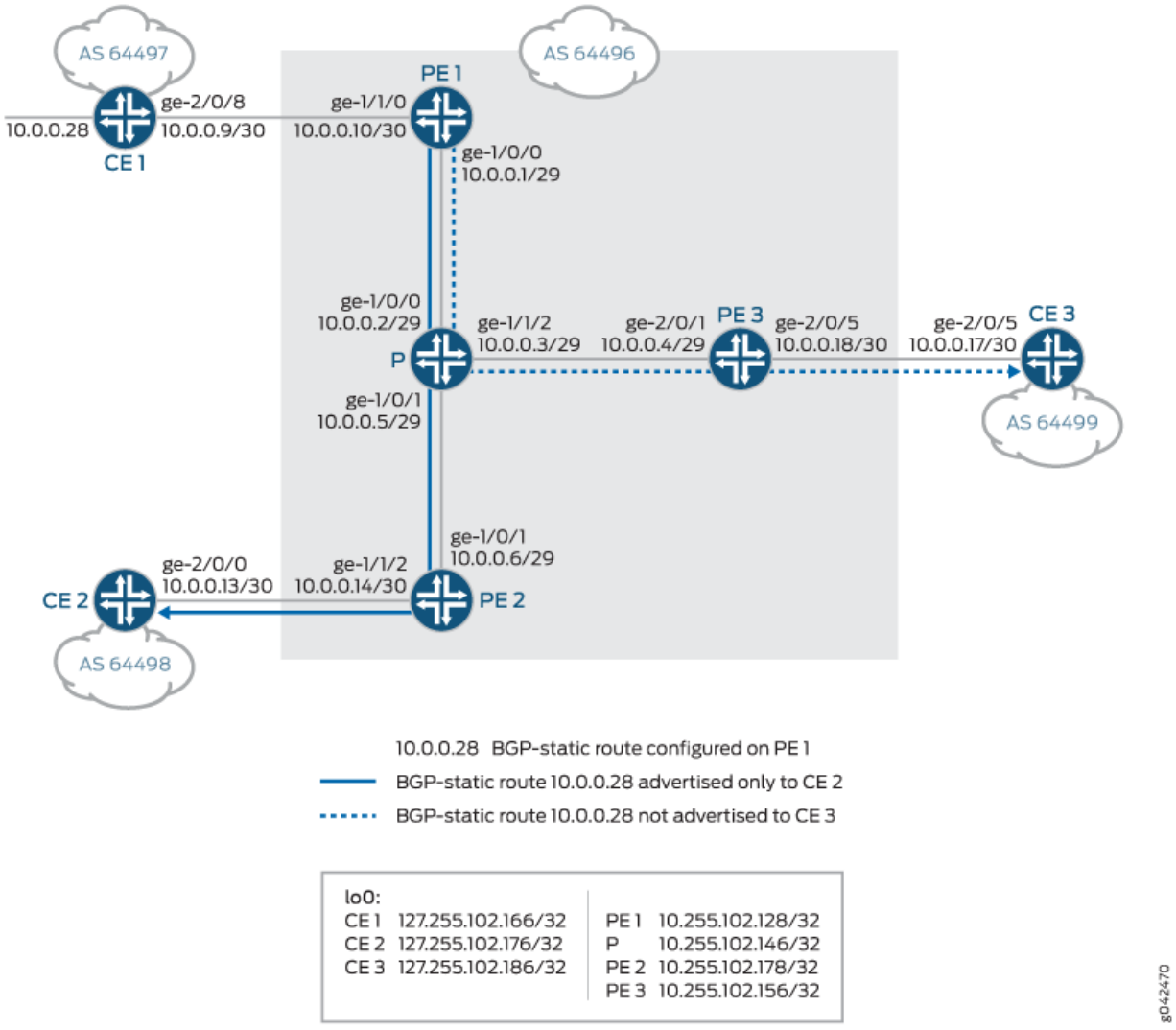
Beginning with Junos OS Release 14.2, you can configure and advertise BGP-static routes in a BGP network. You can advertise a BGP-static route in a BGP network even if it is not the active route for the prefix. BGP-static routes do not flap unless they are deleted manually. You can define a policy that determines which BGP-static routes need to be advertised and included in the advertisements. Peer routers receive advertisements for these BGP-static routes regardless of dynamic routing information learned by the advertising router.

In the sample BGP network, Devices CE1, CE2, and CE3 are directly connected to Routers PE1, PE2, and PE3. Both PE1 and PE2 are connected to Router P. Router P is directly connected to Router PE3. EBGP is configured on the provider edge and customer edge routers. IBGP is configured on directly connected provider edge routers. The IGP protocol IS-IS is configured on all provider routers. Configure a BGP-static route on Router PE1 to ensure that customer route 10.0.0.28 behind CE1 does not flap. Provider Router PE2 is configured to receive the BGP-static route. The objective is to advertise a BGP-static route only to CE2 and not to CE3, and to demonstrate that the configured BGP-static route does not flap.

Topology

Figure 95 on page 1351 shows the sample topology.

Figure 95: Configuring BGP-Static Route



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1352
- Procedure | 1356
- Results | 1358

CLI Quick Configuration

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

Router P

```
set interfaces ge-1/0/0 unit 2 description P->PE1
set interfaces ge-1/0/0 unit 2 family inet address 10.0.0.2/29
set interfaces ge-1/0/0 unit 2 family iso
set interfaces ge-1/0/1 unit 5 description P->PE2
set interfaces ge-1/0/1 unit 5 family inet address 10.0.0.5/29
set interfaces ge-1/0/1 unit 5 family iso
set interfaces ge-1/1/2 unit 3 description P->PE3
set interfaces ge-1/1/2 unit 3 family inet address 10.0.0.3/29
set interfaces ge-1/1/2 unit 3 family iso
set interfaces lo0 unit 0 family inet address 10.255.102.146/32 primary
set interfaces lo0 unit 0 family iso address 49.0001.1720.1600.1050.00
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.102.146
set protocols bgp group ibgp neighbor 10.255.102.128 description PE1
set protocols bgp group ibgp neighbor 10.255.102.178 description PE2
set protocols bgp group ibgp neighbor 10.255.102.156 description PE3
set protocols isis interface ge-1/0/0.2
set protocols isis interface ge-1/0/1.5
set protocols isis interface ge-1/1/2.3
set protocols isis interface lo0.0 passive
set routing-options router-id 10.255.102.146
```

```
set routing-options autonomous-system 64496
```

Router PE1

```
set interfaces ge-1/0/0 unit 1 description PE1->P
set interfaces ge-1/0/0 unit 1 family inet address 10.0.0.1/29
set interfaces ge-1/0/0 unit 1 family iso
set interfaces ge-1/1/0 unit 10 description PE1->CE1
set interfaces ge-1/1/0 unit 10 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 10.255.102.128/32
set interfaces lo0 unit 0 family iso address 49.0001.1720.1600.1010.00
set protocols bgp group ebgp type external
set protocols bgp group ebgp peer-as 64497
set protocols bgp group ebgp neighbor 10.0.0.9 description CE1
set protocols bgp group ebgp neighbor 10.0.0.9 local-address 10.0.0.10
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.102.128
set protocols bgp group ibgp export export-self
set protocols bgp group ibgp neighbor 10.255.102.146 description P
set protocols bgp group ibgp neighbor 10.255.102.178 description PE2
set protocols bgp group ibgp neighbor 10.255.102.178 advertise-bgp-static
set protocols bgp group ibgp neighbor 10.255.102.156 description PE3
set protocols isis interface ge-1/0/0.1
set protocols isis interface lo0.0 passive
set policy-options policy-statement export-self then next-hop self
set routing-options bgp-static route 10.0.0.28/32 preference2 4294967195
set routing-options bgp-static route 10.0.0.28/32 as-path path 64497
set routing-options router-id 10.255.102.128
set routing-options autonomous-system 64496
```

Router PE2

```
set interfaces ge-1/0/1 unit 6 description PE2->P
set interfaces ge-1/0/1 unit 6 family inet address 10.0.0.6/29
set interfaces ge-1/0/1 unit 6 family iso
set interfaces ge-1/1/2 unit 14 description PE2->CE2
set interfaces ge-1/1/2 unit 14 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 10.255.102.178/32
set interfaces lo0 unit 0 family iso address 49.0001.1720.1600.1030.00
set protocols bgp group ebgp type external
```

```
set protocols bgp group ebgp peer-as 64498
set protocols bgp group ebgp neighbor 10.0.0.13 description CE2
set protocols bgp group ebgp neighbor 10.0.0.13 local-address 10.0.0.14
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.102.178
set protocols bgp group ibgp export export-self
set protocols bgp group ibgp neighbor 10.255.102.146 description P
set protocols bgp group ibgp neighbor 10.255.102.128 description PE1
set protocols bgp group ibgp neighbor 10.255.102.156 description PE3
set protocols isis interface ge-1/0/1.6
set protocols isis interface lo0.0 passive
set policy-options policy-statement export-self then next-hop self
set routing-options router-id 10.255.102.178
set routing-options autonomous-system 64496
```

Router PE3

```
set interfaces ge-2/0/1 unit 4 description PE3->P
set interfaces ge-2/0/1 unit 4 family inet address 10.0.0.4/29
set interfaces ge-2/0/5 unit 18 description PE3->CE3
set interfaces ge-2/0/5 unit 18 family inet address 10.0.0.18/30
set interfaces lo0 unit 0 family inet address 10.255.102.156/32
set interfaces lo0 unit 0 family iso address 49.0001.1720.1600.1070.00
set protocols bgp group ebgp type external
set protocols bgp group ebgp peer-as 64499
set protocols bgp group ebgp neighbor 10.0.0.17 description CE3
set protocols bgp group ebgp neighbor 10.0.0.17 local-address 10.0.0.18
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.102.156
set protocols bgp group ibgp export export-self
set protocols bgp group ibgp neighbor 10.255.102.146 description P
set protocols bgp group ibgp neighbor 10.255.102.128 description PE1
set protocols bgp group ibgp neighbor 10.255.102.178 description PE2
set protocols isis interface ge-2/0/1.4
set protocols isis interface lo0.0 passive
set policy-options policy-statement export-self then next-hop self
set routing-options router-id 10.255.102.156
set routing-options autonomous-system 64496
```

Router CE1

```
set interfaces ge-2/0/8 unit 9 description CE1->PE1
set interfaces ge-2/0/8 unit 9 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 127.255.102.166/32
set interfaces lo0 unit 0 family inet address 10.0.0.28/32
set protocols bgp group ebgp type external
set protocols bgp group ebgp export export-direct
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp neighbor 10.0.0.10 description PE1
set protocols bgp group ebgp neighbor 10.0.0.10 local-address 10.0.0.9
set policy-options policy-statement export-direct from protocol direct route-filter 10.0.0.0/29
or longer
set policy-options policy-statement export-direct then accept
set routing-options autonomous-system 64497
```

Router CE2

```
set interfaces ge-2/0/0 unit 13 description CE2->PE2
set interfaces ge-2/0/0 unit 13 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 127.255.102.176/32
set protocols bgp group ebgp type external
set protocols bgp export export-direct
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp neighbor 10.0.0.14 description PE2
set protocols bgp group ebgp neighbor 10.0.0.14 local-address 10.0.0.13
set policy-options policy-statement export-direct from protocol direct route-filter 10.0.0.0/29
or longer
set policy-options policy-statement export-direct then accept
set routing-options router-id 127.255.102.176
set routing-options autonomous-system 64498
```

Router CE3

```
set interfaces ge-2/0/5 unit 17 description CE3->PE3
set interfaces ge-2/0/5 unit 17 family inet address 10.0.0.17/30
set interfaces lo0 unit 0 family inet address 127.255.102.186/32
set protocols bgp group ebgp type external
set protocols bgp export export-direct
set protocols bgp group ebgp peer-as 64496
set protocols bgp group ebgp neighbor 10.0.0.18 description PE3
```

```
set protocols bgp group ebgp neighbor 10.0.0.18 local-address 10.0.0.17
set policy-options policy-statement export-direct from protocol direct route-filter 10.0.0.0/29
or longer
set policy-options policy-statement export-direct then accept
set routing-options router-id 127.255.102.186
set routing-options autonomous-system 64499
```

Procedure

Step-by-Step Procedure

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

To configure Router PE1:

1. Configure the interfaces with IPv4 addresses.

```
[edit interfaces]
user@PE1# set ge-1/0/0 unit 1 description PE1->P
user@PE1# set ge-1/0/0 unit 1 family inet address 10.0.0.1/29
user@PE1# set ge-1/1/0 unit 10 description PE1->CE1
user@PE1# set ge-1/1/0 unit 10 family inet address 10.0.0.10/30
```

2. Enable the IS-IS protocol on interfaces connected to provider routers for learning and exchanging routes learned.

```
[edit interfaces]
user@PE1# set ge-1/0/0 unit 1 family iso
```

3. Configure loopback addresses for inet and IS-IS.

```
[edit interfaces lo0 unit 0]
user@PE1# set family inet address 10.255.102.128/32
user@PE1# set family iso address 49.0001.1720.1600.1010.00
```

4. Configure the IS-IS interfaces.

```
[edit protocols isis]
user@PE1# set interface ge-1/0/0.1
user@PE1# set interface lo0.0 passive
```

5. Configure EBGP.

```
[edit protocols bgp group ebgp]
user@PE1# set type external
user@PE1# set peer-as 64497
user@PE1# set neighbor 10.0.0.9 description CE1
user@PE1# set neighbor 10.0.0.9 local-address 10.0.0.10
```

6. Configure an IBGP neighbor on internal routers connected to the provider network.

```
[edit protocols bgp group ibgp]
user@PE1# set type internal
user@PE1# set local-address 10.255.102.128
user@PE1# set export export-self
user@PE1# set neighbor 10.255.102.146 description P
user@PE1# set neighbor 10.255.102.178 description PE2
user@PE1# set neighbor 10.255.102.156 description PE3
```

7. Configure the BGP static route.

```
[edit routing-options]
user@PE1# set bgp-static route 10.0.0.28/32 preference2 4294967195
user@PE1# set bgp-static route 10.0.0.28/32 as-path path 64497
```

8. Configure the BGP neighbor PE2 to receive BGP-static advertisements.

```
[edit protocols bgp group ibgp neighbor 10.255.102.178]
user@PE1# set advertise-bgp-static
```

9. Define a policy to export routes to the BGP network.

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

10. Apply the policy to the IBGP group.

```
[edit protocols bgp group ibgp]
user@PE1# set export export-self
```

11. Configure the router id and the autonomous system (AS) number.

```
[edit routing-options]
user@PE1# set router-id 10.255.102.128
user@PE1# set autonomous-system 64496
```

Results

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

```
[edit]
user@PE1> show interfaces
ge-1/0/0 {
    unit 1 {
        description PE1->P;
        family inet {
            address 10.0.0.1/29;
        }
        family iso;
    }
    ge-1/1/0 {
        unit 10 {
            description PE1->CE1;
            family inet {
                address 10.0.0.10/30;
            }
        }
```

```
    }
}
lo0 {
    unit 0{
        family inet {
            address 10.255.102.128/32;
        }
        family iso {
            address 49.0001.1720.1600.1010.00;
        }
    }
}
```

```
[edit]
user@PE1> show protocols
bgp {
    group ebgp {
        type external;
        peer-as 64497;
        neighbor 10.0.0.9 {
            description CE1;
            local-address 10.0.0.10;
        }
    }
    group ibgp {
        type internal;
        local-address 10.255.102.128;
        export export-self;
        neighbor 10.255.102.146 {
            description P;
        }
        neighbor 10.255.102.178 {
            description PE2;
            advertise-bgp-static;
        }
        neighbor 10.255.102.156 {
            description PE3;
        }
    }
}
isis {
```

```
    interface ge-1/0/0.1;
    interface lo0.0 {
        passive;
    }
}
```

```
[edit]
user@PE1> show routing-options
bgp-static {
    route 10.0.0.28/32 {
        preference2 4294967195;
        as-path {
            path 64497;
        }
    }
}
router-id 10.255.102.128;
autonomous-system 64496;
```

```
[edit]
user@PE1> show policy-options
policy-statement export-self {
    then {
        next-hop self;
    }
}
```

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

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

Verification

IN THIS SECTION

- Verifying the BGP Neighbors | **1361**

- Verifying BGP Groups | **1365**
- Verifying the Routes | **1366**
- Verifying That the Configured Hosts Receive the BGP-Static Routes | **1367**
- Verifying That the Configured BGP-Static Route Does Not Flap | **1368**

Confirm that the configuration is working properly.

Verifying the BGP Neighbors

Purpose

Verify that BGP is running on the configured interfaces and that the BGP session is active for each neighbor address.

Action

From operational mode, run the **show bgp neighbor** command on Router PE1.

```
user@PE1> show bgp neighbor
Peer: 10.0.0.9+34260 AS 64497    Local: 10.0.0.10+45824 AS 64496
  Description: CE1
  Type: External    State: Established    Flags: <sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: Cease
  Options: <Preference LocalAddress PeerAS Refresh>
  LocalAddress: 10.0.0.10 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 127.255.102.166       Local ID: 10.255.102.128   Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0           Peer index: 0
  BFD: disabled, down
  Local Interface: ge-1/1/0.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
```

```
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 64497)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 14   Sent 13   Checked 4
  Input messages:  Total 249     Updates 2       Refreshes 0     Octets 4764
  Output messages: Total 250     Updates 2       Refreshes 0     Octets 4883

Peer: 10.255.102.146+179 AS 64496  Local: 10.255.102.128+53460 AS 64496
  Description: P
  Type: Internal    State: Established    Flags: <Sync>
Last State: OpenConfirm   Last Event: RecvKeepAlive
Last Error: None
Export: [ export-self ]
Options: <Preference LocalAddress Refresh>
Local Address: 10.255.102.128  Holdtime: 90 Preference: 170
Number of flaps: 0
Peer ID: 10.255.102.146      Local ID: 10.255.102.128     Active Holdtime: 90
Keepalive Interval: 30         Group index: 0         Peer index: 0
BFD: disabled, down
NLRI for restart configured on peer: inet-unicast
NLRI advertised by peer: inet-unicast
NLRI for this session: inet-unicast
Peer supports Refresh capability (2)
Stale routes from peer are kept for: 300
Peer does not support Restarter functionality
Restart flag received from the peer: Notification
NLRI that restart is negotiated for: inet-unicast
NLRI of received end-of-rib markers: inet-unicast
NLRI of all end-of-rib markers sent: inet-unicast
Peer does not support LLGR Restarter functionality
Peer supports 4 byte AS extension (peer-as 64496)
Peer does not support Addpath
Table inet.0 Bit: 10001
  RIB State: BGP restart is complete
```

```
  Send state: in sync
  Active prefixes:              0
  Received prefixes:            0
  Accepted prefixes:            0
  Suppressed due to damping:    0
  Advertised prefixes:          1
Last traffic (seconds): Received 12   Sent 1    Checked 63
Input messages:  Total 246    Updates 1       Refreshes 0     Octets 4678
Output messages: Total 249    Updates 1       Refreshes 0     Octets 4834
Output Queue[0]: 0            (inet.0, inet-unicast)

Peer: 10.255.102.178+53463 AS 64496 Local: 10.255.102.128+179 AS 64496
Description: PE2   Type: Internal    State: Established    Flags: <Synch>
Last State: OpenConfirm   Last Event: RecvKeepAlive
Last Error: None
Export: [ export-self ]
Options:   <Preference LocalAddress Refresh>
Options:   <AdvertiseBGPStatic>
Local Address: 10.255.102.128 Holdtime: 90 Preference: 170
Number of flaps: 0
Peer ID: 10.255.102.178   Local ID: 10.255.102.128     Active Holdtime: 90
Keepalive Interval: 30         Group index: 1    Peer index: 0
BFD: disabled, down
NLRI for restart configured on peer: inet-unicast
NLRI advertised by peer: inet-unicast
NLRI for this session: inet-unicast
Peer supports Refresh capability (2)
Stale routes from peer are kept for: 300
Peer does not support Restarter functionality
Restart flag received from the peer: Notification
NLRI that restart is negotiated for: inet-unicast
NLRI of received end-of-rib markers: inet-unicast
NLRI of all end-of-rib markers sent: inet-unicast
Peer does not support LLGR Restarter functionality
Peer supports 4 byte AS extension (peer-as 64496)
Peer does not support Addpath
Table inet.0 Bit: 10002
  RIB State: BGP restart is complete
  Send state: in sync
  Active prefixes:              1
  Received prefixes:            1
  Accepted prefixes:            1
  Suppressed due to damping:    0
```

```
  Advertised prefixes:          1
Last traffic (seconds): Received 9    Sent 10   Checked 22
Input messages:  Total 247    Updates 2       Refreshes 0     Octets 4777
Output messages: Total 248    Updates 1       Refreshes 0     Octets 4815
Output Queue[0]: 0            (inet.0, inet-unicast)

Peer: 10.255.102.156+179 AS 64496 Local: 10.255.102.128+53462 AS 64496
Description: PE3
Type: Internal    State: Established    Flags:  <Synch>
Last State: OpenConfirm   Last Event: RecvKeepAlive
Last Error: None
Export: [ export-self ]
Options:  <Preference LocalAddress Refresh>
Local Address: 10.255.255.11 Holdtime: 90 Preference: 170
Number of flaps: 0
Peer ID: 10.255.102.156   Local ID: 10.255.102.128     Active Holdtime: 90
Keepalive Interval: 30         Group index: 0    Peer index: 1
BFD: disabled, down
NLRI for restart configured on peer: inet-unicast
NLRI advertised by peer: inet-unicast
NLRI for this session: inet-unicast
Peer supports Refresh capability (2)
Stale routes from peer are kept for: 300
Peer does not support Restarter functionality
Restart flag received from the peer: Notification
NLRI that restart is negotiated for: inet-unicast
NLRI of received end-of-rib markers: inet-unicast
NLRI of all end-of-rib markers sent: inet-unicast
Peer does not support LLGR Restarter functionality
Peer supports 4 byte AS extension (peer-as 64496)
Peer does not support Addpath
Table inet.0 Bit: 10001
  RIB State: BGP restart is complete
  Send state: in sync
  Active prefixes:              1
  Received prefixes:            1
  Accepted prefixes:            1
  Suppressed due to damping:    0
  Advertised prefixes:          1
Last traffic (seconds): Received 21   Sent 10   Checked 10
Input messages:  Total 245    Updates 2       Refreshes 0     Octets 4695
```

```
Output messages: Total 247    Updates 1       Refreshes 0     Octets 4796
Output Queue[0]: 0            (inet.0, inet-unicast)
```

Meaning

The output displays the BGP neighbors of Router PE1 and the configured BGP options such as whether the neighbor is configured to receive BGP-static routes. Router PE2 is configured to receive BGP-static route advertisements.

Verifying BGP Groups

Purpose

Verify that the intended BGP groups or neighbors are configured to receive the BGP-static routes.

Action

From operational mode, run the **show bgp group** command.

```
user@PE1> show bgp group

Group Type: External                              Local AS: 64496
Name: ebgp            Index: 3                  Flags: <Export Eval>
Holdtime: 0 Local AS: 64496 Local System AS: 64496
Total peers: 1        Established: 1
10.0.0.9+179
inet.0: 0/1/1/0

Group Type: Internal    AS: 64496                     Local AS: 64496
Name: ibgp            Index: 0                  Flags: <Export Eval>
Export: [ export-self ]
Options: <AdvertiseBGPStatic>
Holdtime: 0
Total peers: 1        Established: 1
10.255.102.178+179
inet.0: 0/0/0/0

Group Type: Internal    AS: 64496                     Local AS: 64496
Name: ibgp            Index: 0                  Flags: <Export Eval>
Export: [ export-self ]
Holdtime: 0
```

```
Total peers: 2        Established: 2
10.255.102.156+179
10.255.102.146+179
inet.0: 0/3/2/0


Groups: 3  Peers: 4    External: 1    Internal: 3    Down peers: 0   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State
Pending
inet.0                     3          3          0          0          0          0
```

Meaning

The output shows the BGP neighbor that is configured to receive BGP-static advertisements.

Verifying the Routes

Purpose

Verify that the configured BGP-static route is saved in the routing table of the configured BGP neighbors.

Action

From operational mode, run the **show route protocol bgp-static** command to display the routing table.

```
user@PE1> show route protocol bgp-static
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.28/32       *[BGP-Static/4294967292/-101] 00:43:15
                      Discard

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
inet6.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
```

```
User@PE1> show route 10.0.0.28/32
```

```
inet.0: 13 destinations, 14 routes (13 active, 1 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.28/32       *[BGP/170] 00:00:15, localpref 100
                      AS path: 64497 I, validation-state: unverified
                    > to 10.0.0.9 via ge-2/1/8.0
                    [BGP-Static/4294967292/-101] 02:42:51
                      Discard
```

Meaning

The output shows the BGP-static route configured on the device. The active path is learned from CE1, and the BGP-static route is inactive.

Verifying That the Configured Hosts Receive the BGP-Static Routes

Purpose

Verify that the BGP-static route is being advertised to the host configured to receive it.

Action

On Devices CE2 and CE3, from operational mode, run the **show route protocol bgp** command to display the learned routes in the routing table.

```
user@CE2> show route protocol bgp

inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.28/32        *[BGP/170] 01:52:10, localpref 100
                      AS path: 64496 64497 I, validation-state: unverified
                    > to 10.0.0.14 via ge-2/0/0.13
1.0.0.29/32          *[BGP/170] 01:52:06, localpref 100
                      AS path: 64496 64499 I, validation-state: unverified
                    > to 10.0.0.14 via ge-2/0/0.13

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
```

```
user@CE3> show route protocol bgp


inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.0.0.28/32        *[BGP/170] 01:52:19, localpref 100
                      AS path: 64496 64497 I, validation-state: unverified
                    > to 10.0.0.18 via ge-2/0/5.17

1.0.0.29/32        *[BGP/170] 01:52:15, localpref 100
                      AS path: 64496 64498 I, validation-state: unverified
                    > to 10.0.0.18 via ge-2/0/5.17

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
```

Meaning

Both Devices CE2 and CE3 have a route to 10.0.0.28/32. CE2 has received the BGP-static route and CE3 has received a dynamically-learned route, but you cannot tell the difference.

Verifying That the Configured BGP-Static Route Does Not Flap

Purpose

Verify that the BGP-static route does not flap even when the BGP peering session between Router PE1 and Device CE1 goes down.

Action

Deactivate the BGP peering session between Router PE1 and Device CE1. PE1 does not have a dynamically learned route to 10.0.0.28/32, but still has the configured BGP-static route.

```
[edit]
user@PE1# deactivate protocols bgp group ebgp
user@PE1# commit
```

```
user@PE1> show route 10.0.0.28/32
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.28/32         *[BGP-Static/4294967292/-101] 02:46:21
                      Discard
user@CE2> show route protocol bgp
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.28/32        *[BGP/170] 01:52:48, localpref 100
                      AS path: 64496 64497 I, validation-state: unverified
                    > to 10.0.0.18 via ge-2/0/5.17
1.0.0.29/32         *[BGP/170] 01:52:44, localpref 100
                      AS path: 64496 64499 I, validation-state: unverified
                    > to 10.0.0.18 via ge-2/0/5.17

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
user@CE3> show route protocol bgp
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.29/32         *[BGP/170] 01:52:47, localpref 100
                      AS path: 64496 64498 I, validation-state: unverified
                    > to 10.0.0.18 via ge-2/0/5.17

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
```

Meaning

Router PE1 and Device CE2 still have the configured BGP-static route. However, Device CE3 does not have the route to 10.0.0.28/32 because this prefix has flapped. BGP-static routes do not flap unless deleted manually.

SEE ALSO

advertise-bgp-static

bgp-static

Understanding BGP-Static Routes for Preventing Route Flaps | **1347**

Change History Table

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

Release	Description
12.2	Starting in Junos OS Release 12.2, you can apply flap damping at the address family level.

BGP Error Messages

IN THIS SECTION

- Understanding Error Handling for BGP Update Messages | **1370**
- Example: Configuring Error Handling for BGP Update Messages | **1373**

Understanding Error Handling for BGP Update Messages

A BGP message is considered to be malformed when any one of the message attributes is malformed. When a router participating in a BGP session receives a malformed update message, the entire session is reset by default. This is undesirable because update messages with valid routes are also affected. To avoid this undesirable behavior, the error handling for BGP update messages needs to be modified.

To configure error handling for BGP update messages, configure the `bgp-error-tolerance` statement at the `[edit protocols bgp]`, `[edit protocols bgp group group-name]`, or `[edit protocols bgp group group-name neighbor address]` hierarchy level.

```
bgp-error-tolerance {
    malformed-route-limit number;
    malformed-update-log-interval seconds;
    no-malformed-route-limit;
}
```

If an attribute contains attribute flags that conflict with the value of the Attribute Type field, the attribute flags are reset to the correct value and the update message is processed. The value of the Extended Length bit in the attribute flags is unchanged because this value defines whether the attribute length is one or two octets. Hence, the value of the attribute flag affects how the BGP update packet is parsed.

NOTE: There is no explicit specification for the attribute flag value for the path attributes.

Starting in Junos OS Release 24.2R1, BGP error handling is enabled by default. You can still configure sub-options such as, `malformed-route-limit`, `malformed-update-log-interval`, and `no-malformed-route-limit` under this configuration statement. Note that If you delete the `bgp-error-tolerance` statement, the feature still remains enabled and the sub-options are reset to their default values.

Malformed update messages are treated on a case by case basis, depending on the values of the attributes contained in the messages. There are three ways of handling malformed BGP update messages, listed in the decreasing order of severity.

1. **Notification message approach**—The malformed message error is logged locally, an error code update message is sent to the administration of the peer, and the entire BGP session is reset.

   This approach is chosen when:

   - The BGP update message contains the MP reach attribute or the MP unreach attribute.
   - The NLRI field or the BGP update message cannot be parsed correctly because of a mismatch between the attribute length and the value of the attribute length field.

2. **Treat-as-withdraw approach**—All routes within the malformed update message are treated as hidden routes, unless the `keep none` statement is configured, in which case the routes are discarded. In the absence of the `keep none` statement, the number of hidden malformed routes are configured with a limit, which when exceeded discards the routes and prevents any further malformed routes from

being hidden. Junos OS removes the newly received malformed routes when the malformed route limit is reached.

3. **Attribute discard approach**—The malformed attributes in the update message are discarded; however, the message is processed. We do not recommend using this approach if the attributes to be discarded can affect route selection or installation.

> **NOTE**: If an attribute appears more than once in an update message, all occurrences of the attribute, other than the first, will be discarded and the message will be processed.

The BGP update messages are scanned for the following attributes and are treated as malformed based on the values of these attributes:

- **The origin attribute**—Handled by the treat-as-withdraw approach.
- **The AS path attribute**—Handled by the treat-as-withdraw approach.
- **The AS 4 path attribute**—Handled by the attribute discard approach.
- **The aggregator attribute**—Handled by the attribute discard approach.
- **The aggregator 4 attribute**—Handled by the attribute discard approach.
- **The next-hop attribute**—Handled by the treat-as-withdraw approach.
- **The multiple exit discriminator attribute**—Handled by the treat-as-withdraw approach.
- **The local preference attribute**—Handled by the treat-as-withdraw approach.
- **The atomic aggregate attribute**—Handled by the attribute discard approach.
- **The community attribute**—Handled by the treat-as-withdraw approach.
- **The extended community attribute**—Handled by the treat-as-withdraw approach.
- **The originator attribute**—Handled by the treat-as-withdraw approach.
- **The cluster attribute**—Handled by the treat-as-withdraw approach.
- **The PMSI attribute**—Handled by the treat-as-withdraw approach.
- **The MP reach attribute**—Handled by the notification message approach.
- **The MP unreach attribute**—Handled by the notification message approach.
- **The attribute set attribute**—Handled by the treat-as-withdraw approach.
- **The AIGP attribute**—Handled by the treat-as-withdraw approach.

- **Unknown attribute**—If the BGP flag does not indicate that this is an optional attribute, this malformed attribute is handled by the notification message approach.

> **NOTE**: When a BGP update message contains multiple malformed attributes, the most severe approach triggered by one of the attributes is followed.

Here's a sample of the BGP Error Message output:

```
user@R1> show log messages
Sep 18 17:54:13 R1 rpd[86600]: Received malformed update from 10.10.10.2 (External AS 64511)
Sep 18 17:54:13 R1 rpd[86600]: Family inet-unicast, prefix 100.1.1.0/24
Sep 18 17:54:13 R1 rpd[86600]: Malformed Attribute ORIGIN(1) flag 0x40 length 1 error 6
(Unrecognized ORIGIN attribute).
Sep 18 17:54:13 R1 rpd[86600]: Malformed Attribute LOCAL_PREF(5) flag 0x40 length 6 error 5
(Attribute length error).
```

In this example, you see the origin (ORIGIN) and local preference (LOCAL_PREF) malformed attributes.

SEE ALSO

Example: Preventing BGP Session Resets

Example: Configuring Error Handling for BGP Update Messages

IN THIS SECTION

- Requirements | 1374
- Overview | 1374
- Configuration | 1376
- Verification | 1382

This example shows how to configure BGP error handling.

Requirements

Before you begin:

- Configure router interfaces.
- Configure an interior gateway protocol (IGP).
- Configure BGP.
- Configure routing policies.

Overview

IN THIS SECTION

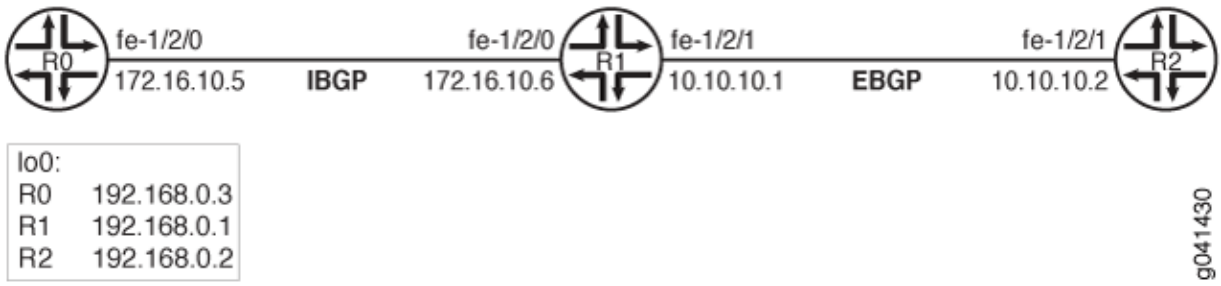
- Topology | **1374**

When a routing device receives an update message with a malformed attribute, the router is required to reset the session. This is specified in RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*. Session resets impact not only routes with the offending attribute, but also other valid routes exchanged over the session. Moreover, this behavior can present a potential security vulnerability in the case of optional transitive attributes. To minimize the impact on routing made by malformed update messages, the Internet draft draft-ietf-idr-error-handling-01.txt, *Revised Error Handling for BGP UPDATE Messages* specifies modifications for handling BGP update message with malformed attributes. The new error handling allows for maintaining the established session and keeping the valid routes exchanged, while removing the routes carried in the malformed UPDATE message.

Topology

In Figure 96 on page 1375, Device R1 has an internal BGP peering session with Device R0, and an external BGP peering session with Device R2.

Figure 96: BGP Error Handling Example Topology



To protect against malformed update messages causing network instability, Device R1 has BGP error handling configured, as shown here:

```
bgp-error-tolerance {
    malformed-update-log-interval 10;
    malformed-route-limit 5;
}
```

By default, a BGP message is considered to be malformed when any one of the message attributes is malformed. When a router participating in a BGP session receives a malformed update message, the entire session is reset. The `bgp-error-tolerance` statement overrides this behavior so that the following BGP error handling is in effect:

- For fatal errors, Junos OS sends a notification message titled Error Code Update Message and resets the BGP session. An error in the MP_{UN}REACH attribute is considered to be fatal. The presence of multiple MP_{UN}REACH attributes in one BGP update is also considered to be a fatal error. Junos OS resets the BGP session if it cannot parse the NLRI field or the BGP update correctly. Failure to parse the BGP update packet can happen when the attribute length does not match the length of the attribute value.

- For some nonfatal errors, Junos OS treats all the routes contained in the malformed BGP update message as withdrawn routes and installs them as hidden, unless the `keep none` statement is included in the BGP is configuration. Junos OS uses this error handling approach for the cases that involve any of the following attributes: ORIGIN, AS_PATH, NEXT_HOP, MULTI_EXIT_DISC, LOCAL_PREF, ORIGINATOR, CLUSTER, ATTRSET, PMSI, Community, and Extended Community. In addition, if any of the mandatory well-known path attributes is missing, Junos OS treats the BGP update as malformed. To limit the memory usage of these malformed hidden routes, Junos OS stops installing new malformed hidden routes after the maximum number of such malformed hidden routes is reached. In this example, the maximum number is set to 5, using the `malformed-route-limit` statement. The default value is 1000. Optionally, you can allow an unlimited number of routes hidden due to malformed attributes. Do this by including the `no-malformed-route-limit` statement.

- For other nonfatal errors, Junos OS discards the malformed path attributes and continues to process the BGP update message. It is unsafe to use this approach on the path attributes that might affect route selection or installation. Junos OS uses this error handling approach for the cases that involve any of the following attributes: ATOMIC_AGGREGATE, AGGREGATOR, AGGREGATOR4, and AS4PATH.

To facilitate troubleshooting of malformed packets, Junos OS logs the error listing the malformed path attribute code, flag, length, information about the peer and family, and the first prefix from the malformed BGP update. Logging of the malformed packets might slow Junos OS performance if a significant number of malformed packets is received in a short time. To limit the performance impact, Junos OS implements an algorithm to log a malformed update, suppress logging for an interval, and log a summary. When the logging suppression timer expires, the software logs the total number of malformed attributes received during the interval. In this example, the timer is set to 10 seconds, using the `malformed-update-log-interval` statement. The default value is 300 seconds(5 minutes).

"CLI Quick Configuration" on page 1376 shows the configuration for all of the devices in Figure 96 on page 1375.

The section "No Link Title" on page 1378 describes the steps on Device R1.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1376**
- Procedure | **1378**

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 R0

```
set  interfaces fe-1/2/0 unit 0 description to-R1
set  interfaces fe-1/2/0 unit 0 family inet address 172.16.10.5/30
set  interfaces lo0 unit 0 family inet address 192.168.0.3/32
set  protocols bgp group internal-peers type internal
set  protocols bgp group internal-peers local-address 192.168.0.3
```

```
set  protocols bgp group internal-peers export local-direct
set  protocols bgp group internal-peers neighbor 192.168.0.1
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  policy-options policy-statement local-direct from protocol [local direct]
set  policy-options policy-statement local-direct then accept
set  routing-options autonomous-system 64510
set  routing-options router-id 192.168.0.3
```

Device R1

```
set  interfaces fe-1/2/1 unit 0 description to-R2
set  interfaces fe-1/2/1 unit 0 family inet address 10.10.10.1/30
set  interfaces fe-1/2/0 unit 0 description to-R0
set  interfaces fe-1/2/0 unit 0 family inet address 172.16.10.6/30
set  interfaces lo0 unit 0 family inet address 192.168.0.1/32
set  protocols bgp bgp-error-tolerance malformed-update-log-interval 10
set  protocols bgp bgp-error-tolerance malformed-route-limit 5
set  protocols bgp group internal-peers type internal
set  protocols bgp group internal-peers local-address 192.168.0.1
set  protocols bgp group internal-peers export local-direct
set  protocols bgp group internal-peers neighbor 192.168.0.3
set  protocols bgp group external-peers type external
set  protocols bgp group external-peers export local-direct
set  protocols bgp group external-peers peer-as 64511
set  protocols bgp group external-peers neighbor 10.10.10.2
set  protocols ospf area 0.0.0.0 interface fe-1/2/1.0
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  policy-options policy-statement local-direct from protocol [local direct]
set  policy-options policy-statement local-direct then accept
set  routing-options autonomous-system 64510
set  routing-options router-id 192.168.0.1
```

Device R2

```
set  interfaces fe-1/2/1 unit 0 description to-R1
set  interfaces fe-1/2/1 unit 0 family inet address 10.10.10.2/30
set  interfaces lo0 unit 0 family inet address 192.168.0.2/32
set  protocols bgp group external-peers type external
set  protocols bgp group external-peers export local-direct
```

```
set  protocols bgp group external-peers peer-as 64510
set  protocols bgp group external-peers neighbor 10.10.10.1
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  policy-options policy-statement local-direct from protocol [local direct]
set  policy-options policy-statement local-direct then accept
set  routing-options autonomous-system 64511
set  routing-options router-id 192.168.10.2
```

Procedure

Step-by-Step Procedure

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

To configure the BGP error handling:

1. Configure the router interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/1 unit 0 description to-R2
user@R1# set fe-1/2/1 unit 0 family inet address 10.10.10.1/30
user@R1# set fe-1/2/0 unit 0 description to-R0
user@R1# set fe-1/2/0 unit 0 family inet address 172.16.10.6/30
user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure an interior gateway protocol (IGP), such as OSPF or IS-IS.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/1.0
user@R1# set interface fe-1/2/0.0
user@R1# set interface lo0.0 passive
```

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

```
[edit routing-options]
user@R1# set autonomous-system 64510
user@R1# set router-id 192.168.0.1
```

4. Configure the routing policy.

```
[edit policy-options policy-statement local-direct]
user@R1# set from protocol [local direct]
user@R1# set then accept
```

5. Configure the EBGP session.

```
[edit protocols bgp group external-peers]
user@R1# set type external
user@R1# set export local-direct
user@R1# set peer-as 64511
user@R1# set neighbor 10.10.10.2
```

6. Configure the IBGP sessions.

```
[edit protocols bgp group internal-peers]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set export local-direct
user@R1# set neighbor 192.168.0.3
```

7. Enable BGP error tolerance.

```
[edit protocols bgp]
user@R1# set bgp-error-tolerance
```

8. (Optional) Configure the log interval.

```
[edit protocols bgp bgp-error-tolerance]
user@R1# set malformed-update-log-interval 10
```

9. (Optional) Configure a limit for the number of hidden routes to store.

```
[edit protocols bgp bgp-error-tolerance]
user@R1# set malformed-route-limit 5
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, 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@R1# show interfaces
fe-1/2/0 {
    unit 0 {
        description to-R0;
        family inet {
            address 172.16.10.6/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        description to-R2;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
```

```
    }
}
```

```
user@R1# show protocols
bgp {
    bgp-error-tolerance {
        malformed-update-log-interval 10;
        malformed-route-limit 5;
    }
    group internal-peers {
        type internal;
        local-address 192.168.0.1;
        export local-direct;
        neighbor 192.168.0.3;
    }
    group external-peers {
        type external;
        export local-direct;
        peer-as 64511;
        neighbor 10.10.10.2;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.0;
        interface fe-1/2/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement local-direct {
    from protocol [local direct];
```

```
    then accept;
}
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 64510;
```

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

Verification

IN THIS SECTION

- Checking the BGP Neighbor Sessions | **1382**
- Checking Hidden Routes | **1384**
- Verifying the Source of the Hidden Routes | **1386**

Confirm that the configuration is working properly.

Checking the BGP Neighbor Sessions

Purpose

Verify that BGP error tolerance is enabled, and display the counters related to malformed path attributes.

Action

```
user@R1# show bgp neighbor
Peer: 10.10.10.2+50058 AS 64511 Local: 10.10.10.1+179 AS 64510
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ local-direct ]
  Options: <Preference PeerAS Refresh>
  Holdtime: 90 Preference: 170
  Number of flaps: 0
```

```
  Malformed attributes    log interval: 10   route limit: 5
    Attribute:           ORIGIN(1) Last Received: 0 Total Received: 3
    Attribute:       LOCAL_PREF(5) Last Received: 0 Total Received: 2
  Peer ID: 192.168.10.2    Local ID: 192.168.10.1      Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/2/1.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 64511)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            3
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 25   Sent 17   Checked 73
  Input messages:  Total 2702   Updates 10      Refreshes 0     Octets 51652
  Output messages: Total 2701   Updates 6       Refreshes 0     Octets 51571
  Output Queue[0]: 0

Peer: 192.168.10.3+179 AS 64510 Local: 192.168.10.1+51127 AS 64510
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ local-direct ]
  Options: <Preference LocalAddress Refresh>
  Local Address: 192.168.10.1 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Malformed attributes    log interval: 10   route limit: 5
  Peer ID: 192.168.10.3    Local ID: 192.168.10.1      Active Holdtime: 90
  Keepalive Interval: 30         Group index: 1    Peer index: 0
  BFD: disabled, down
```

```
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-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-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 64510)
  Peer does not support Addpath
  Table inet.0 Bit: 10001
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            3
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          2
  Last traffic (seconds): Received 5    Sent 24   Checked 51
  Input messages:  Total 417    Updates 3       Refreshes 0     Octets 8006
  Output messages: Total 421    Updates 2       Refreshes 0     Octets 8136
  Output Queue[0]: 0
```

Meaning

The Malformed attributes field shows that error tolerance is enabled. The log interval and route limit fields display the configured values.

The attribute counters show that on the EBGP connection, several malformed attributes were received from Device R2.

Checking Hidden Routes

Purpose

View information about hidden routes and learn why they are hidden.

Action

```
user@R1> show route hidden detail
inet.0: 42 destinations, 45 routes (36 active, 0 holddown, 6 hidden)
10.0.0.0/32 (1 entry, 0 announced)
         BGP
                Next hop type: Router
                Address: 0x93d8b0c
                Next-hop reference count: 5
                Source: 10.10.10.2
                Next hop type: Router, Next hop index: 782
                Next hop: via fe-1/2/1.0, selected
                Session Id: 0x1
                State: <Hidden  Ext>
                Local AS:     1 Peer AS:     1
                Age: 5:32       Metric2: 1
                Validation State: unverified
                Task: BGP_1.10.10.5.62+56218
                AS path: I (MalformedAttr)
                Router ID: 192.168.0.2

10.0.0.1/32 (1 entry, 0 announced)
         BGP
                Next hop type: Router
                Address: 0x93d8b0c
                Next-hop reference count: 5
                Source: 10.10.10.2
                Next hop type: Router, Next hop index: 782
                Next hop: via fe-1/2/1.0, selected
                Session Id: 0x1
                Indirect next hop: 953c000 - INH Session ID: 0x3
                State: <Hidden Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 5:32       Metric2: 1
                Validation State: unverified
                Task: BGP_1.10.10.5.62+56218
                AS path: I (MalformedAttr)
                Router ID: 192.168.0.2
```

Meaning

The malformed hidden routes are marked with MalformedAttr in the AS path field.

You can remove the hidden routes by running the `clear bgp neighbor 10.10.10.2 malformed-route` command.

Verifying the Source of the Hidden Routes

Purpose

View information about hidden routes and learn why they are hidden.

Action

```
user@R1> show route receive-protocol bgp 10.10.10.2 detail hidden
inet.0: 42 destinations, 45 routes (36 active, 0 holddown, 6 hidden)
  10.0.0.0/32 (1 entry, 0 announced)
     Nexthop: 10.10.10.2
     Localpref: 100
     AS path: I (MalformedAttr)

  10.0.0.1/32 (1 entry, 0 announced)
     Nexthop: 10.10.10.2
     Localpref: 100
     AS path: I (MalformedAttr)
```

Meaning

Junos OS displays MalformedAttr in the AS path field in the output of the `show route receive-protocol bgp 10.10.10.2 detail hidden` command.

You can remove the hidden routes by running the `clear bgp neighbor 10.10.10.2 malformed-route` command.

SEE ALSO

- Example: Preventing BGP Session Resets
- Examples: Configuring BGP Flap Damping

BFD for BGP Sessions

IN THIS SECTION

- Understanding BFD for BGP | **1387**
- Example: Configuring BFD on Internal BGP Peer Sessions | **1388**
- Understanding BFD Authentication for BGP | **1401**
- Example: Configuring BFD Authentication for BGP | **1403**
- Platform-Specific BFD for BGP Behavior | **1407**

Understanding BFD for BGP

The Bidirectional Forwarding Detection (BFD) protocol is a simple hello mechanism that detects failures in a network. Hello packets are sent at a specified, regular interval. A neighbor failure is detected when the routing device stops receiving a reply after a specified interval. BFD works with a wide variety of network environments and topologies. The failure detection timers for BFD have shorter time limits than default failure detection mechanisms for BGP, so they provide faster detection.

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific BFD for BGP Behavior" on page 1407 section for notes related to your platform.

NOTE: Configuring both BFD and graceful restart for BGP on the same device is counterproductive. When an interface goes down, BFD detects this instantly, stops traffic forwarding and the BGP session goes down whereas graceful restart forwards traffic despite the interface failure, this behavior might cause network issues. Hence we do not recommend configuring both BFD and graceful restart on the same device.

The BFD failure detection timers can be adjusted to be faster or slower. The lower the BFD failure detection timer value, the faster the failure detection and vice versa. For example, the timers can adapt to a higher value if the adjacency fails (that is, the timer detects failures more slowly). Or a neighbor can negotiate a higher value for a timer than the configured value. The timers adapt to a higher value when a BFD session flap occurs more than three times in a span of 15 seconds (15000 milliseconds). A back-off algorithm increases the receive (Rx) interval by two if the local BFD instance is the reason for the session flap. The transmission (Tx) interval is increased by two if the remote BFD instance is the reason for the

session flap. You can use the `clear bfd adaptation` command to return BFD interval timers to their configured values. The `clear bfd adaptation` command is hitless, meaning that the command does not affect traffic flow on the routing device.

SEE ALSO

> *Enabling Dedicated and Real-Time BFD*

Example: Configuring BFD on Internal BGP Peer Sessions

IN THIS SECTION

- Requirements | **1388**
- Overview | **1388**
- Configuration | **1390**
- Verification | **1396**

This example shows how to configure internal BGP (IBGP) peer sessions with the Bidirectional Forwarding Detection (BFD) protocol to detect failures in a network.

Requirements

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

Overview

The minimum configuration to enable BFD on IBGP sessions is to include the `bfd-liveness-detection minimum-interval` statement in the BGP configuration of all neighbors participating in the BFD session. The `minimum-interval` statement specifies the minimum transmit and receive intervals for failure detection. Specifically, this value represents the minimum interval after which the local routing device transmits hello packets as well as the minimum interval that the routing device expects to receive a reply from a neighbor with which it has established a BFD session. You can configure a value from 1 through 255,000 milliseconds.

Optionally, you can specify the minimum transmit and receive intervals separately using the `transmit-interval minimum-interval` and `minimum-receive-interval` statements. For information about these and other optional BFD configuration statements, see `bfd-liveness-detection`.

NOTE: BFD is an intensive protocol that consumes system resources. Specifying a minimum interval for BFD less than 100 milliseconds for Routing Engine-based sessions and less than 10 milliseconds for distributed BFD sessions can cause undesired BFD flapping.

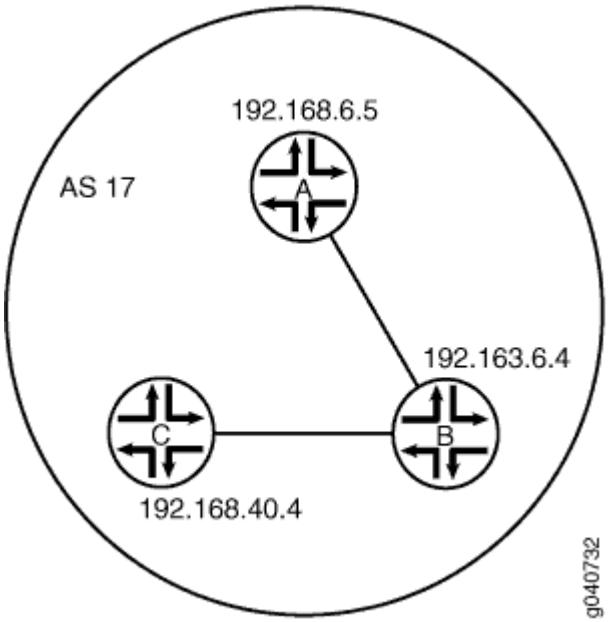
Depending on your network environment, these additional recommendations might apply:

- To prevent BFD flapping during the general Routing Engine switchover event, specify a minimum interval of 5000 milliseconds for Routing Engine-based sessions. This minimum value is required because, during the general Routing Engine switchover event, processes such as RPD, MIBD, and SNMPD utilize CPU resources for more than the specified threshold value. Hence, BFD processing and scheduling is affected because of this lack of CPU resources.
- For BFD sessions to remain up during the dual chassis cluster control link scenario, when the first control link fails, specify the minimum interval of 6000 milliseconds to prevent the LACP from flapping on the secondary node for Routing Engine-based sessions.
- For large-scale network deployments with a large number of BFD sessions, specify a minimum interval of 300 milliseconds for Routing Engine-based sessions and 100 milliseconds for distributed BFD sessions.
- For very large-scale network deployments with a large number of BFD sessions, contact Juniper Networks customer support for more information.
- For BFD sessions to remain up during a Routing Engine switchover event when nonstop active routing (NSR) is configured, specify a minimum interval of 2500 milliseconds for Routing Engine-based sessions. For distributed BFD sessions with NSR configured, the minimum interval recommendations are unchanged and depend only on your network deployment.

BFD is supported on the default routing instance (the main router), routing instances, and logical systems. This example shows BFD on logical systems.

Figure 97 on page 1390 shows a typical network with internal peer sessions.

Figure 97: Typical Network with IBGP Sessions



Configuration

IN THIS SECTION

- CLI Quick Configuration | 1390
- Configuring Device A | 1392

CLI Quick Configuration

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

Device A

```
set logical-systems A interfaces lt-1/2/0 unit 1 description to-B
set logical-systems A interfaces lt-1/2/0 unit 1 encapsulation ethernet
set logical-systems A interfaces lt-1/2/0 unit 1 peer-unit 2
set logical-systems A interfaces lt-1/2/0 unit 1 family inet address 10.10.10.1/30
set logical-systems A interfaces lo0 unit 1 family inet address 192.168.6.5/32
```

```
set logical-systems A protocols bgp group internal-peers type internal
set logical-systems A protocols bgp group internal-peers traceoptions file bgp-bfd
set logical-systems A protocols bgp group internal-peers traceoptions flag bfd detail
set logical-systems A protocols bgp group internal-peers local-address 192.168.6.5
set logical-systems A protocols bgp group internal-peers export send-direct
set logical-systems A protocols bgp group internal-peers bfd-liveness-detection minimum-interval
1000
set logical-systems A protocols bgp group internal-peers neighbor 192.163.6.4
set logical-systems A protocols bgp group internal-peers neighbor 192.168.40.4
set logical-systems A protocols ospf area 0.0.0.0 interface lo0.1 passive
set logical-systems A protocols ospf area 0.0.0.0 interface lt-1/2/0.1
set logical-systems A policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems A policy-options policy-statement send-direct term 2 then accept
set logical-systems A routing-options router-id 192.168.6.5
set logical-systems A routing-options autonomous-system 17
```

Device B

```
set logical-systems B interfaces lt-1/2/0 unit 2 description to-A
set logical-systems B interfaces lt-1/2/0 unit 2 encapsulation ethernet
set logical-systems B interfaces lt-1/2/0 unit 2 peer-unit 1
set logical-systems B interfaces lt-1/2/0 unit 2 family inet address 10.10.10.2/30
set logical-systems B interfaces lt-1/2/0 unit 5 description to-C
set logical-systems B interfaces lt-1/2/0 unit 5 encapsulation ethernet
set logical-systems B interfaces lt-1/2/0 unit 5 peer-unit 6
set logical-systems B interfaces lt-1/2/0 unit 5 family inet address 10.10.10.5/30
set logical-systems B interfaces lo0 unit 2 family inet address 192.163.6.4/32
set logical-systems B protocols bgp group internal-peers type internal
set logical-systems B protocols bgp group internal-peers local-address 192.163.6.4
set logical-systems B protocols bgp group internal-peers export send-direct
set logical-systems B protocols bgp group internal-peers bfd-liveness-detection minimum-interval
1000
set logical-systems B protocols bgp group internal-peers neighbor 192.168.40.4
set logical-systems B protocols bgp group internal-peers neighbor 192.168.6.5
set logical-systems B protocols ospf area 0.0.0.0 interface lo0.2 passive
set logical-systems B protocols ospf area 0.0.0.0 interface lt-1/2/0.2
set logical-systems B protocols ospf area 0.0.0.0 interface lt-1/2/0.5
set logical-systems B policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems B policy-options policy-statement send-direct term 2 then accept
set logical-systems B routing-options router-id 192.163.6.4
set logical-systems B routing-options autonomous-system 17
```

Device C

```
set logical-systems C interfaces lt-1/2/0 unit 6 description to-B
set logical-systems C interfaces lt-1/2/0 unit 6 encapsulation ethernet
set logical-systems C interfaces lt-1/2/0 unit 6 peer-unit 5
set logical-systems C interfaces lt-1/2/0 unit 6 family inet address 10.10.10.6/30
set logical-systems C interfaces lo0 unit 3 family inet address 192.168.40.4/32
set logical-systems C protocols bgp group internal-peers type internal
set logical-systems C protocols bgp group internal-peers local-address 192.168.40.4
set logical-systems C protocols bgp group internal-peers export send-direct
set logical-systems C protocols bgp group internal-peers bfd-liveness-detection minimum-interval
1000
set logical-systems C protocols bgp group internal-peers neighbor 192.163.6.4
set logical-systems C protocols bgp group internal-peers neighbor 192.168.6.5
set logical-systems C protocols ospf area 0.0.0.0 interface lo0.3 passive
set logical-systems C protocols ospf area 0.0.0.0 interface lt-1/2/0.6
set logical-systems C policy-options policy-statement send-direct term 2 from protocol direct
set logical-systems C policy-options policy-statement send-direct term 2 then accept
set logical-systems C routing-options router-id 192.168.40.4
set logical-systems C routing-options autonomous-system 17
```

Configuring Device A

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 A:

1. Set the CLI to Logical System A.

```
user@host> set cli logical-system A
```

2. Configure the interfaces.

```
[edit interfaces lt-1/2/0 unit 1]
user@host:A# set description to-B
user@host:A# set encapsulation ethernet
```

```
user@host:A# set peer-unit 2
user@host:A# set family inet address 10.10.10.1/30
[edit interfaces lo0 unit 1]
user@host:A# set family inet address 192.168.6.5/32
```

3. Configure BGP.

   The `neighbor` statements are included for both Device B and Device C, even though Device A is not directly connected to Device C.

```
[edit protocols bgp group internal-peers]
user@host:A# set type internal
user@host:A# set local-address 192.168.6.5
user@host:A# set export send-direct
user@host:A# set neighbor 192.163.6.4
user@host:A# set neighbor 192.168.40.4
```

4. Configure BFD.

```
[edit protocols bgp group internal-peers]
user@host:A# set bfd-liveness-detection minimum-interval 1000
```

   You must configure the same minimum interval on the connecting peer.

5. (Optional) Configure BFD tracing.

```
[edit protocols bgp group internal-peers]
user@host:A# set traceoptions file bgp-bfd
user@host:A# set traceoptions flag bfd detail
```

6. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@host:A# set interface lo0.1 passive
user@host:A# set interface lt-1/2/0.1
```

7. Configure a policy that accepts direct routes.

Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 2]
user@host:A# set from protocol direct
user@host:A# set then accept
```

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

```
[edit routing-options]
user@host:A# set router-id 192.168.6.5
user@host:A# set autonomous-system 17
```

9. If you are done configuring the device, enter `commit` from configuration mode.
   Repeat these steps to configure Device B and Device C.

Results

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

```
user@host:A# show interfaces
lt-1/2/0 {
    unit 1 {
        description to-B;
        encapsulation ethernet;
        peer-unit 2;
        family inet {
            address 10.10.10.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 192.168.6.5/32;
        }
```

```
    }
}
```

```
user@host:A# show policy-options
policy-statement send-direct {
    term 2 {
        from protocol direct;
        then accept;
    }
}
```

```
user@host:A# show protocols
bgp {
    group internal-peers {
        type internal;
        traceoptions {
            file bgp-bfd;
            flag bfd detail;
        }
        local-address 192.168.6.5;
        export send-direct;
        bfd-liveness-detection {
            minimum-interval 1000;
        }
        neighbor 192.163.6.4;
        neighbor 192.168.40.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface lt-1/2/0.1;
```

```
    }
}
```

```
user@host:A# show routing-options
router-id 192.168.6.5;
autonomous-system 17;
```

Verification

IN THIS SECTION

- Verifying That BFD Is Enabled | **1396**
- Verifying That BFD Sessions Are Up | **1397**
- Viewing Detailed BFD Events | **1398**
- Viewing Detailed BFD Events After Deactivating and Reactivating a Loopback Interface | **1399**

Confirm that the configuration is working properly.

Verifying That BFD Is Enabled

Purpose

Verify that BFD is enabled between the IBGP peers.

Action

From operational mode, enter the `show bgp neighbor` command. You can use the `| match bfd` filter to narrow the output.

```
user@host:A> show bgp neighbor | match bfd
Options: <BfdEnabled>
  BFD: enabled, up
  Trace file: /var/log/A/bgp-bfd size 131072 files 10
  Options: <BfdEnabled>
```

```
  BFD: enabled, up
  Trace file: /var/log/A/bgp-bfd size 131072 files 10
```

Meaning

The output shows that Logical System A has two neighbors with BFD enabled. When BFD is not enabled, the output displays `BFD: disabled, down`, and the `<BfdEnabled>` option is absent. If BFD is enabled and the session is down, the output displays `BFD: enabled, down`. The output also shows that BFD-related events are being written to a log file because trace operations are configured.

Verifying That BFD Sessions Are Up

Purpose

Verify that the BFD sessions are up, and view details about the BFD sessions.

Action

From operational mode, enter the `show bfd session extensive` command.

```
user@host:A> show bfd session extensive
                                                Detect   Transmit
Address                 State     Interface     Time     Interval  Multiplier
192.163.6.4             Up                      3.000     1.000        3
 Client BGP, TX interval 1.000, RX interval 1.000
 Session up time 00:54:40
 Local diagnostic None, remote diagnostic None
 Remote state Up, version 1
 Logical system 12, routing table index 25
 Min async interval 1.000, min slow interval 1.000
 Adaptive async TX interval 1.000, RX interval 1.000
 Local min TX interval 1.000, minimum RX interval 1.000, multiplier 3
 Remote min TX interval 1.000, min RX interval 1.000, multiplier 3
 Local discriminator 10, remote discriminator 9
 Echo mode disabled/inactive
 Multi-hop route table 25, local-address 192.168.6.5

                                                Detect   Transmit
Address                 State     Interface     Time     Interval  Multiplier
192.168.40.4            Up                      3.000     1.000        3
 Client BGP, TX interval 1.000, RX interval 1.000
```

```
  Session up time 00:48:03
  Local diagnostic None, remote diagnostic None
  Remote state Up, version 1
  Logical system 12, routing table index 25
  Min async interval 1.000, min slow interval 1.000
  Adaptive async TX interval 1.000, RX interval 1.000
  Local min TX interval 1.000, minimum RX interval 1.000, multiplier 3
  Remote min TX interval 1.000, min RX interval 1.000, multiplier 3
  Local discriminator 14, remote discriminator 13
  Echo mode disabled/inactive
  Multi-hop route table 25, local-address 192.168.6.5

2 sessions, 2 clients
Cumulative transmit rate 2.0 pps, cumulative receive rate 2.0 pps
```

Meaning

The `TX interval 1.000, RX interval 1.000` output represents the setting configured with the `minimum-interval` statement. All of the other output represents the default settings for BFD. To modify the default settings, include the optional statements under the `bfd-liveness-detection` statement.

Viewing Detailed BFD Events

Purpose

View the contents of the BFD trace file to assist in troubleshooting, if needed.

Action

From operational mode, enter the `file show /var/log/A/bgp-bfd` command.

```
user@host:A> file show /var/log/A/bgp-bfd
Aug 15 17:07:25 trace_on: Tracing to "/var/log/A/bgp-bfd" started
Aug 15 17:07:26.492190 bgp_peer_init: BGP peer 192.163.6.4 (Internal AS 17) local address
192.168.6.5 not found.  Leaving peer idled
Aug 15 17:07:26.493176 bgp_peer_init: BGP peer 192.168.40.4 (Internal AS 17) local address
192.168.6.5 not found.  Leaving peer idled
Aug 15 17:07:32.597979 task_connect: task BGP_17.192.163.6.4+179 addr 192.163.6.4+179: No route
to host
Aug 15 17:07:32.599623 bgp_connect_start: connect 192.163.6.4 (Internal AS 17): No route to host
Aug 15 17:07:36.869394 task_connect: task BGP_17.192.168.40.4+179 addr 192.168.40.4+179: No
```

```
route to host
Aug 15 17:07:36.870624 bgp_connect_start: connect 192.168.40.4 (Internal AS 17): No route to host
Aug 15 17:08:04.599220 task_connect: task BGP_17.192.163.6.4+179 addr 192.163.6.4+179: No route
to host
Aug 15 17:08:04.601135 bgp_connect_start: connect 192.163.6.4 (Internal AS 17): No route to host
Aug 15 17:08:08.869717 task_connect: task BGP_17.192.168.40.4+179 addr 192.168.40.4+179: No
route to host
Aug 15 17:08:08.869934 bgp_connect_start: connect 192.168.40.4 (Internal AS 17): No route to host
Aug 15 17:08:36.603544 advertising receiving-speaker only capabilty to neighbor 192.163.6.4
(Internal AS 17)
Aug 15 17:08:36.606726 bgp_read_message: 192.163.6.4 (Internal AS 17): 0 bytes buffered
Aug 15 17:08:36.609119 Initiated BFD session to peer 192.163.6.4 (Internal AS 17):
address=192.163.6.4 ifindex=0 ifname=(none) txivl=1000 rxivl=1000 mult=3 ver=255
Aug 15 17:08:36.734033 advertising receiving-speaker only capabilty to neighbor 192.168.40.4
(Internal AS 17)
Aug 15 17:08:36.738436 Initiated BFD session to peer 192.168.40.4 (Internal AS 17):
address=192.168.40.4 ifindex=0 ifname=(none) txivl=1000 rxivl=1000 mult=3 ver=255
Aug 15 17:08:40.537552 BFD session to peer 192.163.6.4 (Internal AS 17) up
Aug 15 17:08:40.694410 BFD session to peer 192.168.40.4 (Internal AS 17) up
```

Meaning

Before the routes are established, the `No route to host` message appears in the output. After the routes are established, the last two lines show that both BFD sessions come up.

Viewing Detailed BFD Events After Deactivating and Reactivating a Loopback Interface

Purpose

Check to see what happens after bringing down a router or switch and then bringing it back up. To simulate bringing down a router or switch, deactivate the loopback interface on Logical System B.

Action

1. From configuration mode, enter the `deactivate logical-systems B interfaces lo0 unit 2 family inet` command.

```
user@host:A# deactivate logical-systems B interfaces lo0 unit 2 family inet
user@host:A# commit
```

2. From operational mode, enter the `file show /var/log/A/bgp-bfd` command.

```
user@host:A> file show /var/log/A/bgp-bfd
...
Aug 15 17:20:55.995648 bgp_read_v4_message:9747: NOTIFICATION received from 192.163.6.4
(Internal AS 17): code 6 (Cease) subcode 6 (Other Configuration Change)
Aug 15 17:20:56.004508 Terminated BFD session to peer 192.163.6.4 (Internal AS 17)
Aug 15 17:21:28.007755 task_connect: task BGP_17.192.163.6.4+179 addr 192.163.6.4+179: No
route to host
Aug 15 17:21:28.008597 bgp_connect_start: connect 192.163.6.4 (Internal AS 17): No route to
host
```

3. From configuration mode, enter the `activate logical-systems B interfaces lo0 unit 2 family inet` command.

```
user@host:A# activate logical-systems B interfaces lo0 unit 2 family inet
user@host:A# commit
```

4. From operational mode, enter the `file show /var/log/A/bgp-bfd` command.

```
user@host:A> file show /var/log/A/bgp-bfd
...
Aug 15 17:25:53.623743 advertising receiving-speaker only capabilty to neighbor 192.163.6.4
(Internal AS 17)
Aug 15 17:25:53.631314 Initiated BFD session to peer 192.163.6.4 (Internal AS 17):
address=192.163.6.4 ifindex=0 ifname=(none) txivl=1000 rxivl=1000 mult=3 ver=255
Aug 15 17:25:57.570932 BFD session to peer 192.163.6.4 (Internal AS 17) up
```

SEE ALSO

Example: Configuring BFD Authentication for BGP

Understanding BFD Authentication for BGP

IN THIS SECTION

- BFD Authentication Algorithms | 1401
- Security Authentication Keychains | 1402
- Strict Versus Loose Authentication | 1402

Bidirectional Forwarding Detection protocol (BFD) enables rapid detection of communication failures between adjacent systems. By default, authentication for BFD sessions is disabled. However, when you run BFD over Network Layer protocols, the risk of service attacks can be significant. We strongly recommend using authentication if you are running BFD over multiple hops or through insecure tunnels. Junos OS supports authentication for BFD sessions running over BGP. BFD authentication is not supported on MPLS OAM sessions. BFD authentication is only supported in the Canada and United States version of the Junos OS image and is not available in the export version.

You authenticate BFD sessions by specifying an authentication algorithm and keychain, and then associating that configuration information with a security authentication keychain using the keychain name.

The following sections describe the supported authentication algorithms, security keychains, and level of authentication that can be configured:

BFD Authentication Algorithms

Junos OS supports the following algorithms for BFD authentication:

- **simple-password**—Plain-text password. One to 16 bytes of plain text are used to authenticate the BFD session. One or more passwords can be configured. This method is the least secure and should be used only when BFD sessions are not subject to packet interception.
- **keyed-md5**—Keyed Message Digest 5 hash algorithm for sessions with transmit and receive intervals greater than 100 ms. To authenticate the BFD session, keyed MD5 uses one or more secret keys (generated by the algorithm) and a sequence number that is updated periodically. With this method, packets are accepted at the receiving end of the session if one of the keys matches and the sequence number is greater than or equal to the last sequence number received. Although more secure than a simple password, this method is vulnerable to replay attacks. Increasing the rate at which the sequence number is updated can reduce this risk.

- **meticulous-keyed-md5**—Meticulous keyed Message Digest 5 hash algorithm. This method works in the same manner as keyed MD5, but the sequence number is updated with every packet. Although more secure than keyed MD5 and simple passwords, this method might take additional time to authenticate the session.
- **keyed-sha-1**—Keyed Secure Hash Algorithm I for sessions with transmit and receive intervals greater than 100 ms. To authenticate the BFD session, keyed SHA uses one or more secret keys (generated by the algorithm) and a sequence number that is updated periodically. The key is not carried within the packets. With this method, packets are accepted at the receiving end of the session if one of the keys matches and the sequence number is greater than the last sequence number received.
- **meticulous-keyed-sha-1**—Meticulous keyed Secure Hash Algorithm I. This method works in the same manner as keyed SHA, but the sequence number is updated with every packet. Although more secure than keyed SHA and simple passwords, this method might take additional time to authenticate the session.

NOTE: *Nonstop active routing* (NSR) is not supported with meticulous-keyed-md5 and meticulous-keyed-sha-1 authentication algorithms. BFD sessions using these algorithms might go down after a switchover.

Security Authentication Keychains

The security authentication keychain defines the authentication attributes used for authentication key updates. When the security authentication keychain is configured and associated with a protocol through the keychain name, authentication key updates can occur without interrupting routing and signaling protocols.

The authentication keychain contains one or more keychains. Each keychain contains one or more keys. Each key holds the secret data and the time at which the key becomes valid. The algorithm and keychain must be configured on both ends of the BFD session, and they must match. Any mismatch in configuration prevents the BFD session from being created.

BFD allows multiple clients per session, and each client can have its own keychain and algorithm defined. To avoid confusion, we recommend specifying only one security authentication keychain.

Strict Versus Loose Authentication

By default, strict authentication is enabled and authentication is checked at both ends of each BFD session. Optionally, to smooth migration from nonauthenticated sessions to authenticated sessions, you can configure *loose checking*. When loose checking is configured, packets are accepted without authentication being checked at each end of the session. This feature is intended for transitional periods only.

SEE ALSO

bfd-liveness-detection

Junos OS Administration Library for Routing Devices

CLI Explorer

Example: Configuring BFD on Internal BGP Peer Sessions | **1388**

Example: Configuring BFD Authentication for BGP

IN THIS SECTION

- Configuring BFD Authentication Parameters | **1403**
- Viewing Authentication Information for BFD Sessions | **1405**

Beginning with Junos OS Release 9.6, you can configure authentication for BFD sessions running over BGP. Only three steps are needed to configure authentication on a BFD session:

1. Specify the BFD authentication algorithm for the BGP protocol.
2. Associate the authentication keychain with the BGP protocol.
3. Configure the related security authentication keychain.

The following sections provide instructions for configuring and viewing BFD authentication on BGP:

Configuring BFD Authentication Parameters

BFD authentication can be configured for the entire BGP protocol, or a specific BGP group, neighbor, or routing instance.

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 Junos OS CLI User Guide.

To configure BFD authentication:

1. Specify the algorithm (**keyed-md5**, **keyed-sha-1**, **meticulous-keyed-md5**, **meticulous-keyed-sha-1**, or **simple-password**) to use.

```
[edit]
user@host# set protocols bgp bfd-liveness-detection authentication algorithm keyed-sha-1
user@host# set protocols bgp group bgp-gr1 bfd-liveness-detection authentication algorithm
keyed-sha-1
user@host# set protocols bgp group bgp-gr1 neighbor 10.10.10.7 bfd-liveness-detection
authentication algorithm keyed-sha-1
```

NOTE: Nonstop active routing is not supported with meticulous-keyed-md5 and meticulous-keyed-sha-1 authentication algorithms. BFD sessions using these algorithms might go down after a switchover.

2. Specify the keychain to be used to associate BFD sessions on BGP with the unique security authentication keychain attributes.

   The keychain name you specify must match a keychain name configured at the `[edit security authentication key-chains]` hierarchy level.

```
[edit]
user@host# set protocols bgp bfd-liveness-detection authentication keychain bfd-bgp
user@host# set protocols bgp group bgp-gr1 bfd-liveness-detection authentication keychain bfd-
bgp
user@host# set protocols bgp group bgp-gr1 neighbor 10.10.10.7 bfd-liveness-detection
authentication keychain bfd-bgp
```

NOTE: The algorithm and keychain must be configured on both ends of the BFD session, and they must match. Any mismatch in configuration prevents the BFD session from being created.

3. Specify the unique security authentication information for BFD sessions:

   - The matching keychain name as specified in Step 2.
   - At least one key, a unique integer between **0** and **63**. Creating multiple keys allows multiple clients to use the BFD session.
   - The secret data used to allow access to the session.

- The time at which the authentication key becomes active, in the format *yyyy-mm-dd.hh:mm:ss*.

```
[edit security]
user@host# set authentication-key-chains key-chain bfd-bgp key 53 secret $ABC123$ABC123 start-
time 2009-06-14.10:00:00
```

4. (Optional) Specify loose authentication checking if you are transitioning from nonauthenticated sessions to authenticated sessions.

```
[edit]
user@host# set protocols bgp bfd-liveness-detection authentication loose-check
user@host# set protocols bgp group bgp-gr1 bfd-liveness-detection authentication loose-check
user@host# set protocols bgp group bgp-gr1 neighbor 10.10.10.7 bfd-liveness-detection
authentication loose-check
```

5. (Optional) View your configuration using the `show bfd session detail` or `show bfd session extensive` command.
6. Repeat these steps to configure the other end of the BFD session.

NOTE: BFD authentication is only supported in the Canada and United States version of the Junos OS image and is not available in the export version.

Viewing Authentication Information for BFD Sessions

You can view the existing BFD authentication configuration using the `show bfd session detail` and `show bfd session extensive` commands.

The following example shows BFD authentication configured for the `bgp-gr1` BGP group. It specifies the keyed SHA-1 authentication algorithm and a keychain name of `bfd-bgp`. The authentication keychain is configured with two keys. Key **1** contains the secret data "**$ABC123$ABC123**" and a start time of June 1, 2009, at 9:46:02 AM PST. Key **2** contains the secret data "**$ABC123$ABC123**" and a start time of June 1, 2009, at 3:29:20 PM PST.

```
[edit protocols bgp]
group bgp-gr1 {
    bfd-liveness-detection {
        authentication {
            algorithm keyed-sha-1;
            key-chain bfd-bgp;
        }
    }
```

```
}
[edit security]
authentication key-chains {
    key-chain bfd-bgp {
        key 1 {
            secret “$ABC123$ABC123”;
            start-time “2009-6-1.09:46:02 -0700”;
        }
        key 2 {
            secret “$ABC123$ABC123”;
            start-time “2009-6-1.15:29:20 -0700”;
        }
    }
}
```

If you commit these updates to your configuration, you see output similar to the following. In the output for the `show bfd session detail` command, **Authenticate** is displayed to indicate that BFD authentication is configured. For more information about the configuration, use the `show bfd session extensive` command. The output for this command provides the keychain name, the authentication algorithm and mode for each client in the session, and the overall BFD authentication configuration status, keychain name, and authentication algorithm and mode.

show bfd session detail

```
user@host# show bfd session detail

                                                Detect   Transmit
Address                 State     Interface     Time     Interval  Multiplier
192.0.2.2                Up        ge-0/1/5.0     0.900     0.300        3
 Client BGP, TX interval 0.300, RX interval 0.300, Authenticate
 Session up time 3d 00:34
 Local diagnostic None, remote diagnostic NbrSignal
 Remote state Up, version 1
 Replicated
```

show bfd session extensive

```
user@host# show bfd session extensive
                                                Detect   Transmit
Address                 State     Interface     Time     Interval  Multiplier
192.0.2.2                Up        ge-0/1/5.0     0.900     0.300        3
 Client BGP, TX interval 0.300, RX interval 0.300, Authenticate
```

```
      keychain bfd-bgp, algo keyed-sha-1, mode strict
Session up time 00:04:42
Local diagnostic None, remote diagnostic NbrSignal
Remote state Up, version 1
Replicated
Min async interval 0.300, min slow interval 1.000
Adaptive async TX interval 0.300, RX interval 0.300
Local min TX interval 0.300, minimum RX interval 0.300, multiplier 3
Remote min TX interval 0.300, min RX interval 0.300, multiplier 3
Local discriminator 2, remote discriminator 2
Echo mode disabled/inactive
Authentication enabled/active, keychain bfd-bgp, algo keyed-sha-1, mode strict
```

RELATED DOCUMENTATION

- Understanding BFD Authentication for BGP | **1401**
- *bfd-liveness-detection*
- Example: Configuring BFD for BGP
- Junos OS Administration Library for Routing Devices
- CLI Explorer

Platform-Specific BFD for BGP Behavior

IN THIS SECTION

- Platform-Specific BFD for BGP Behavior | **1408**

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for your platform:

Platform-Specific BFD for BGP Behavior

Platform	Difference
ACX Series	• The `bfd-liveness-detection authentication` configuration statement option is not supported on ACX7000 Series devices.
EX Series	• EX4600 switches do not support minimum interval values of less than 1 second.
QFX Series	• QFX5000 Series switches do not support minimum interval values of less than 1 second.

Change History Table

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

Release	Description
15.1X49-D100	Starting with Junos OS Release 15.1X49-D100, SRX340, SRX345, and SRX1500 devices support dedicated BFD.
15.1X49-D100	Starting with Junos OS Release 15.1X49-D100, SRX300 and SRX320 devices support real-time BFD.
11.2	In Junos OS Release 11.2 and later, BFD supports IPv6 interfaces with BGP.
9.1	In Junos OS Release 9.1 through Junos OS Release 11.1, BFD supports IPv6 interfaces in static routes only.
8.3	In Junos OS Release 8.3 and later, BFD is supported on internal BGP (IBGP) and multihop external BGP (EBGP) sessions as well as on single-hop EBGP sessions.

12

CHAPTER

Configuring BGP-Based VPN

IN THIS CHAPTER

- BGP-Based VPN | **1410**
- BGP accept-own Community | **1423**

BGP-Based VPN

IN THIS SECTION

- Understanding Carrier-of-Carriers VPNs | 1410
- Understanding Interprovider and Carrier-of-Carriers VPNs | 1412
- Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service | 1413

Understanding Carrier-of-Carriers VPNs

IN THIS SECTION

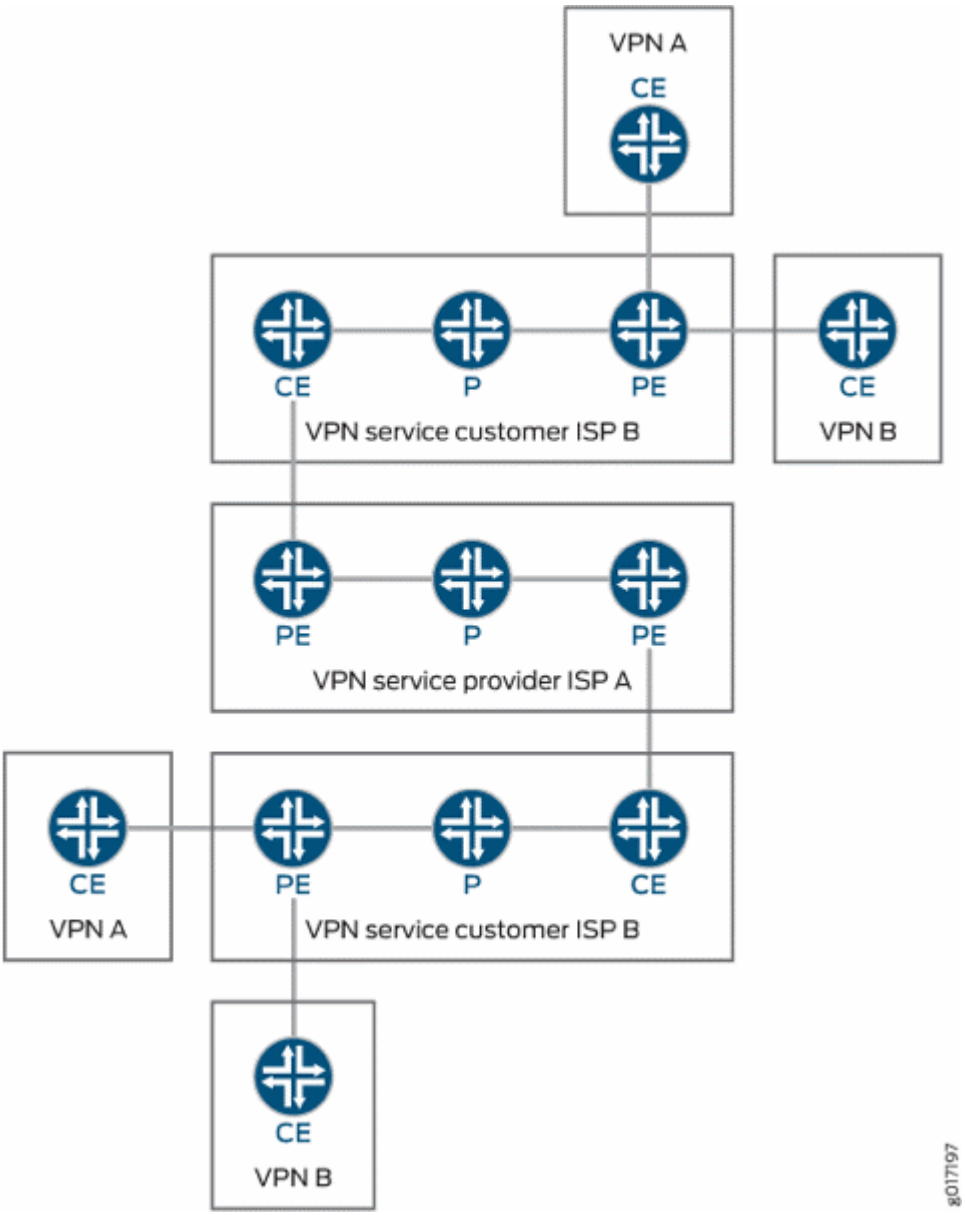
- Internet Service Provider as the Customer | 1411
- VPN Service Provider as the Customer | 1412

The customer of a VPN service provider might be a service provider for the end customer. The following are the two main types of carrier-of-carriers VPNs (as described in RFC 4364:

- "Internet Service Provider as the Customer" on page 1411—The VPN customer is an ISP that uses the VPN service provider's network to connect its geographically disparate regional networks. The customer does not have to configure MPLS within its regional networks.
- "VPN Service Provider as the Customer" on page 1412—The VPN customer is itself a VPN service provider offering VPN service to its customers. The carrier-of-carriers VPN service customer relies on the backbone VPN service provider for inter-site connectivity. The customer VPN service provider is required to run MPLS within its regional networks.

Figure 98 on page 1411 illustrates the network architecture used for a carrier-of-carriers VPN service.

Figure 98: Carrier-of-Carriers VPN Architecture



This topic covers the following:

Internet Service Provider as the Customer

In this type of carrier-of-carriers VPN configuration, ISP A configures its network to provide Internet service to ISP B. ISP B provides the connection to the customer wanting Internet service, but the actual Internet service is provided by ISP A.

This type of carrier-of-carriers VPN configuration has the following characteristics:

- The carrier-of-carriers VPN service customer (ISP B) does not need to configure MPLS on its network.
- The carrier-of-carriers VPN service provider (ISP A) must configure MPLS on its network.
- MPLS must also be configured on the CE routers and PE routers connected together in the carrier-of-carriers VPN service customer's and carrier-of-carriers VPN service provider's networks.

VPN Service Provider as the Customer

A VPN service provider can have customers that are themselves VPN service providers. In this type of configuration, also called a hierarchical or recursive VPN, the customer VPN service provider's VPN-IPv4 routes are considered external routes, and the backbone VPN service provider does not import them into its VRF table. The backbone VPN service provider imports only the customer VPN service provider's internal routes into its VRF table.

The similarities and differences between interprovider and carrier-of-carriers VPNs are shown in Table 12 on page 1412.

Table 12: Comparison of Interprovider and Carrier-of-Carriers VPNs

Feature	ISP Customer	VPN Service Provider Customer
Customer edge device	AS border router	PE router
IBGP sessions	Carry IPv4 routes	Carry external VPN-IPv4 routes with associated labels
Forwarding within the customer network	MPLS is optional	MPLS is required

Support for VPN service as the customer is supported on QFX10000 switches starting with Junos OS Release 17.1R1.

Understanding Interprovider and Carrier-of-Carriers VPNs

All interprovider and carrier-of-carriers VPNs share the following characteristics:

- Each interprovider or carrier-of-carriers VPN customer must distinguish between internal and external customer routes.

- Internal customer routes must be maintained by the VPN service provider in its PE routers.
- External customer routes are carried only by the customer's routing platforms, not by the VPN service provider's routing platforms.

The key difference between interprovider and carrier-of-carriers VPNs is whether the customer sites belong to the same AS or to separate ASs:

- *Interprovider VPNs*—The customer sites belong to different ASs. You need to configure EBGP to exchange the customer's external routes.
- *Understanding Carrier-of-Carriers VPNs*—The customer sites belong to the same AS. You need to configure IBGP to exchange the customer's external routes.

In general, each service provider in a VPN hierarchy is required to maintain its own internal routes in its P routers, and the internal routes of its customers in its PE routers. By recursively applying this rule, it is possible to create a hierarchy of VPNs.

The following are definitions of the types of PE routers specific to interprovider and carrier-of-carriers VPNs:

- The AS border router is located at the AS border and handles traffic leaving and entering the AS.
- The end PE router is the PE router in the customer VPN; it is connected to the CE router at the end customer's site.

Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service

IN THIS SECTION

- Configuring the Carrier-of-Carriers Customer's PE Router | **1414**
- Configuring the Carrier-of-Carriers Customer's CE Router (or switch) | **1417**
- Configuring the Provider's PE Router or Switch | **1420**

You can configure a carrier-of-carriers VPN service for customers who want VPN service.

To configure the routers (or switches) in the customer's and provider's networks to enable carrier-of-carriers VPN service, perform the steps in the following sections:

Configuring the Carrier-of-Carriers Customer's PE Router

The carrier-of-carriers customer's PE router (or switch) is connected to the end customer's CE router (or switch).

The following sections describe how to configure the carrier-of-carriers customer's PE router (or switch):

Configuring MPLS

To configure MPLS on the carrier-of-carriers customer's PE router (or switch), include the `mpls` statement:

```
mpls {
    interface interface-name;
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

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

Configuring BGP

Include the `labeled-unicast` statement in the configuration for the IBGP session to the carrier-of-carriers customer's CE router (or switch)), and include the `family-inet-vpn` statement in the configuration for the IBGP session to the carrier-of-carriers PE router (or switch) on the other side of the network:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        neighbor address {
            family inet {
                labeled-unicast;
                resolve-vpn;
            }
        }
    }
    neighbor address {
        family inet-vpn {
            any;
```

```
        }
    }
}
```

You can include these statements at the following hierarchy levels:

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

Configuring OSPF

To configure OSPF on the carrier-of-carriers customer's PE router (or switch), include the ospf statement:

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

You can include this statement at the following hierarchy levels:

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

Configuring LDP

To configure LDP on the carrier-of-carriers customer's PE router (or switch), include the ldp statement:

```
ldp {
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

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

Configuring VPN Service in the Routing Instance

To configure VPN service for the end customer's CE router (or switch) on the carrier-of-carriers customer's PE router (or switch), include the following statements:

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

You can include these statements at the following hierarchy levels:

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

Configuring Policy Options

To configure policy options to import and export routes to and from the end customer's CE router (or switch), include the `policy-statement` and `community` statements:

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

```
policy-statement policy-name {
    term term-name {
        from protocol bgp;
        then {
            community add community-name;
            accept;
        }
    }
    term term-name {
        then reject;
    }
}
community community-name members value;
```

You can include these statements at the following hierarchy levels:

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

Configuring the Carrier-of-Carriers Customer's CE Router (or switch)

The carrier-of-carriers customer's CE router (or switch) connects to the provider's PE router (or switch). Complete the instructions in the following sections to configure the carrier-of-carriers customers' CE router (or switch):

Configuring MPLS

In the MPLS configuration for the carrier-of-carriers customer's CE router (or switch), include the interfaces to the provider's PE router (or switch) and to a P router (or switch) in the customer's network:

```
mpls {
    traffic-engineering bgp-igp;
    interface interface-name;
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

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

Configuring BGP

In the BGP configuration for the carrier-of-carriers customer's CE router (or switch), configure a group that includes the `labeled-unicast` statement to extend VPN service to the PE router (or switch)connected to the end customer's CE router (or switch):

```
bgp {
    group group-name {
        type internal;
        local-address address;
        neighbor address {
            family inet {
                labeled-unicast;
            }
        }
    }
}
```

You can include the `bgp` statement at the following hierarchy levels:

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

To configure a group to send labeled internal routes to the provider's PE router (or switch), include the `bgp` statement:

```
bgp {
    group group-name {
        export internal;
        peer-as as-number;
        neighbor address {
            family inet {
                labeled-unicast;
            }
        }
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`

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

Configuring OSPF and LDP

To configure OSPF and LDP on the carrier-of-carriers customer's CE router (or switch), include the ospf and ldp statements:

```
ospf {
    area area-id {
        interface interface-name {
            passive;
        }
        interface interface-name;
    }
}
ldp {
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

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

Configuring Policy Options

To configure the policy options on the carrier-of-carriers customer's CE router (or switch), include the policy-statement statement:

```
policy-statement policy-statement-name {
    term term-name {
        from protocol [ ospf direct ldp ];
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

You can include this statement at the following hierarchy levels:

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

Configuring the Provider's PE Router or Switch

The carrier-of-carriers provider's PE routers (or switches) connect to the carrier customer's CE routers (or switches) . Complete the instructions in the following sections to configure the provider's PE router (or switch):

Configuring MPLS

In the MPLS configuration, specify at least two interfaces—one to the customer's CE router (or switch)and one to connect to the provider's PE router (or switch)on the other side of the provider's network:

```
interface interface-name;
interface interface-name;
```

You can include these statements at the following hierarchy levels:

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

Configuring a PE-to-PE BGP Session

To configure a PE-to-PE BGP session on the provider's PE routers (or switches) to allow VPN-IPv4 routes to pass between the PE routers (or switches, include the bgp statement:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        family inet-vpn {
            any;
        }
        neighbor address;
    }
}
```

You can include this statement at the following hierarchy levels:

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

Configuring IS-IS and LDP

To configure IS-IS and LDP on the provider's PE routers (or switches), include the isis and ldp statements:

```
isis {
    interface interface-name;
    interface interface-name {
        passive;
    }
}
ldp {
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

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

Configuring Policy Options

To configure policy statements on the provider's PE router (or switch) to export routes to and import routes from the carrier customer's network, include the policy-statement and community statements:

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

```
policy-statement statement-name {
    term term-name {
        from protocol bgp;
        then {
            community add community-name;
            accept;
        }
    }
    term term-name {
        then reject;
    }
}
community community-name members value;
```

You can include these statements at the following hierarchy levels:

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

Configuring a Routing Instance to Send Routes to the CE Router

To configure the routing instance on the provider's PE router (or switch) to send labeled routes to the carrier customer's CE router (or switch), include the following statements:

```
instance-type vrf;
interface interface-name;
route-distinguisher value;
vrf-import policy-name;
vrf-export policy-name;
protocols {
    bgp {
        group group-name {
            peer-as as-number;
            neighbor address {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
}
```

You can include these statements at the following hierarchy levels:

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

SEE ALSO

MPLS Feature Support on QFX Series and EX4600 Switches

Understanding Interprovider and Carrier-of-Carriers VPNs

Change History Table

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

Release	Description
17.1R1	Support for VPN service as the customer is supported on QFX10000 switches starting with Junos OS Release 17.1R1.

BGP accept-own Community

IN THIS SECTION

- Understanding BGP accept-own Community Attribute | **1423**
- Configure BGP accept-own Community | **1425**

Understanding BGP accept-own Community Attribute

IN THIS SECTION

- Benefits of BGP accept-own Community Attribute | **1424**

Overview of BGP accept-own Community Attribute | 1424

Benefits of BGP accept-own Community Attribute

- Helps to leak routes from one VPN instance to another on the same provider edge(PE) device.
- Convenient in VPN deployment scenarios where BGP route reflector controls how a route originated from one VRF is imported to another VRF on the same provider edge (PE) device.
- Enhances interoperability while replacing non-Junos routers with Junos routers on customer networks.

Overview of BGP accept-own Community Attribute

Per the standard BGP specification, a BGP speaker rejects routes received with the following attributes:

- The originator id set to itself.
- The nexthop attribute same as that of the receiver's own IP address.

Starting in Junos OS Release 21.4R1, MX480 and MX960 routers accept BGP routes with the `accept-own`community, defined by *RFC 7611, BGP ACCEPT_OWN Community Attribute*. The feature enables Juniper routers to accept routes whose `ORIGINATOR_ID` or `NEXT_HOP` value matches that of the receiving BGP speaker. For example, when a provider edge (PE) device advertises routes with the route distinguisher of a source VRF, the route reflector attaches the `accept-own` community, adds more route targets, and re-advertises the routes back to the originator. The provider edge (PE) device can then import the routes into the other destination VRFs, excluding its own.

NOTE: We support `accept-own` configuration only for `inet-vpn unicast` and `inet6-vpn unicast` address families.

Per RFC 7611, routes attached with ACCEPT_OWN community should be preferred over routes that do not have the community after the LOCAL_PREF comparison is done in the BGP decision process.

SEE ALSO

accept-own

Configure BGP accept-own Community

Before you configure accept-own community, make sure you:

1. Configure the device interfaces.
2. Configure router ID and autonomous system number.
3. Configure OSPF or any other IGP protocol.
4. Configure LDP.
5. Configure MPLS.

The sections shows how to enable the routers to accept routes with the `accept-own` community from a route reflector.

1. Configure an internal BGP connection.

```
user@PE#
set protocols bgp group group type internal
set protocols bgp group group local-address local-address
set protocols bgp group group neighbor neighbor-address
```

2. Configure policy options to export and accept static routes, add them to a community with a specified route target. Configure a unique route target for the added community.

```
user@PE#
set policy-options policy-statement exportpolicy from protocol static
set policy-options policy-statement exportpolicy then community add community
set policy-options policy-statement exportpolicy then accept

set policy-options policy-statement importpolicy from community community
set policy-options policy-statement importpolicy then accept
set policy-options community community members targetcommunity:ID1
```

3. Configure two routing instances with unique route distinguishers and route target to create two VRFs with route export and import. Configure a static route with a next-hop address.

```
user@PE#
set routing-instances vrf1 instance-type vrf
set routing-instances vrf1 route-distinguisher route-distinguisher
set routing-instances vrf1 routing-options static route route next-hop address
set routing-instances vrf1 interface interface
```

```
set routing-instances vrf1 vrf-import importpolicy
set routing-instances vrf1 vrf-export exportpolicy
set routing-instances vrf2 instance-type vrf
set routing-instances vrf2 route-distinguisher route-distinguisher2
set routing-instances vrf2 vrf-target targetcommunity:ID2
```

4. Configure the following statement to enable `accept-own` community.

```
user@PE#
set protocols bgp group group neighbor address family inet-vpn unicast accept-own
```

5. Enter `commit` to commit the configuration.

SEE ALSO

accept-own

13

CHAPTER

Monitoring and Troubleshooting

IN THIS CHAPTER

- BGP Monitoring Protocol | **1428**
- Troubleshooting Network Issues | **1456**
- Troubleshooting BGP Sessions | **1476**

BGP Monitoring Protocol

IN THIS SECTION

- Monitoring BGP Routing Information | 1428
- Understanding the BGP Monitoring Protocol | 1429
- Configuring BGP Monitoring Protocol Version 3 | 1430
- Configuring BGP Monitoring Protocol to Run Over a Different Routing Instance | 1431
- Example: Configuring the BGP Monitoring Protocol | 1434
- Understanding Trace Operations for BGP Protocol Traffic | 1437
- Example: Viewing BGP Trace Files on Logical Systems | 1440
- Example: Tracing Global Routing Protocol Operations | 1447
- Tracing BMP Operations | 1454

Monitoring BGP Routing Information

IN THIS SECTION

- Purpose | 1428
- Action | 1428

Purpose

Use the monitoring functionality to monitor BGP routing information on the routing device.

Action

To view BGP routing information in the CLI, enter the following commands:

- `show bgp summary`

- `show bgp neighbor`

SEE ALSO

> *show bgp neighbor*
>
> *show bgp summary*

Understanding the BGP Monitoring Protocol

The BGP Monitoring Protocol (BMP) is a protocol to allow a monitoring station to receive routes from a BGP-enabled device. The monitoring station receives all routes, not just the active routes. BMP uses route monitoring messages (which are essentially encapsulated BGP update messages) and a few other message types for statistics and state changes. All messages flow from the router to the monitoring station.

NOTE: When an interface is disabled, the BMP that monitors the TCP session, is shut down for 240 seconds (4 minutes). This is an expected behavior.

The data is collected from the `Adjacency-RIB-In` routing tables. The `Adjacency-RIB-In` tables are the pre-policy tables, meaning that the routes in these tables have not been filtered or modified by routing policies.

NOTE: The `Local-RIB` tables are the post-policy tables.

Starting in Junos OS Release 22.4R1, you can configure a policy to monitor routing information bases (RIBs) of type virtual router and virtual routing and forwarding (VRF). You can specify two separate sets of RIBs in the BGP Monitoring Protocol (BMP), one for monitoring and the other for reporting. With this feature, BMP can filter traffic based on the routes and routing instances.

SEE ALSO

> Example: Configuring the BGP Monitoring Protocol | **1434**
>
> Configuring BGP Monitoring Protocol Version 3 | **1430**

Configuring BGP Monitoring Protocol Version 3

BGP Monitoring Protocol (BMP) allows the Junos OS to send the BGP route information from the router to a monitoring application on a separate device. The monitoring application is called the BMP monitoring station or BMP station. To deploy BMP in your network, you need to configure BMP on each router and you also need to configure at least one BMP station. This procedure describes how to configure BMP on a router.

You can specify these settings for all BMP stations by configuring the statements described here at the `[edit routing-options bmp]` hierarchy level. You can also configure settings for specific BMP stations by configuring these statements at the `[edit routing-options bmp station station-name]` hierarchy level.

The following procedure describes how to configure BMP version 3 on the router:

1. Specify the memory limit for the BMP monitoring station by configuring the `memory limit` statement. The value must be in bytes.

```
memory limit bytes;
```

2. Specify the name or address for the BMP monitoring station by configuring the `station-address` statement. You can specify one or the other but not both. The address must be a valid IPv4 or IPv6 address.

```
station-address (ip-address | station-name);
```

3. Specify the port number for the BMP monitoring station by configuring the `station-port` statement.

```
station-port port-number;
```

4. Configure how often statistics messages are sent to the BMP monitoring station by specifying the number of seconds between message transmissions using `statistics-timeout` statement. If you configure a value of 0, no statistics messages are sent.

```
statistics-timeout seconds;
```

SEE ALSO

Example: Configuring Router Authentication for BGP | 1236

Configuring BGP Monitoring Protocol to Run Over a Different Routing Instance

IN THIS SECTION

- Configuring a Nondefault Routing Instance for BMP | 1431
- Configuring mgmt_junos for BMP | 1432

Starting in Junos OS Release 18.3R1, you can specify which routing instance you want the BGP Monitoring Protocol (BMP) to use. Prior to Junos OS Release 18.3R1, you had to use the default routing instance. By default, in Junos OS, the management Ethernet interface (usually named fxp0 or em0) provides the out-of-band management network for the device. There is no clear separation between either out-of-band management traffic and in-band protocol control traffic, or user traffic at the routing-instance or routing-table level. Instead, all traffic is handled through the default routing instance, giving rise to concerns over security, performance, and how to troubleshoot.

Starting with Junos OS Release 17.3R1, you can configure the management interface in a non-default virtual routing and forwarding (VRF) instance, the mgmt_junos routing instance. Once you configure this management routing instance as described in Configuring the mgmt_junos Routing Instance, management traffic no longer has to share a routing table (that is, the default.inet.0 table) with other control or protocol traffic in the system. But it is only as of Junos OS Release 18.3R1 that you can use this non-default management instance for BMP. You can also use any configured routing instance for BMP. It no longer has to be the default routing instance.

Configuring a Nondefault Routing Instance for BMP

To modify the routing instance that BMP uses, you must configure the BMP station and the connection mode, which is either passive or active. In active mode, the router attempts to start the TCP connection with the BMP station. In passive mode the router waits for the BMP station to initiate the TCP session. You also must configure a port and the station address.

NOTE: To use a non-default routing instance, you must configure it under the `[edit routing-instances]` hierarchy level.

To configure a non-default routing instance for BMP:

1. Configure the routing instance under the `edit routing-instances` hierarchy level.

```
user@host# set routing-instances routing-instance-name description description
```

2. Configure the routing instance for the BMP routing instance.

```
user@host# set routing-options bmp station station-name routing-instance routing-instance-name
```

3. Configure the connection mode.

```
user@host# set routing-options bmp station station-name connection-mode (active | passive)
```

- If you configure passive mode, configure the following additional statements:

```
set routing-options bmp station station-name local-address ip-address
set routing-options bmp station station-name local-port port-number
set routing-options bmp station station-name station-address ip-address
```

- If you configure active mode, configure at least the following additional statements:

```
set routing-options bmp station station-name station-address ip-address
set routing-options bmp station station-name station-port port-number
```

Configuring mgmt_junos for BMP

To modify the routing instance that BMP uses, you must configure the BMP station and the connection mode, which is either passive or active. In active mode, the router attempts to start the TCP connection with the BMP station. In passive mode the router waits for the BMP station to initiate the TCP session. You also must configure a port and the station address.

NOTE: To use the management routing instance, you must configure it under the `[edit routing-instances]` hierarchy level, and you must enable it using the `management-instance` configuration statement.

To configure mgmt_junos as the routing-instance for BMP:

1. Configure the non-default management routing instance.

```
user@host# set system management-instance
```

2. Configure the routing instance under the `edit routing-instances` hierarchy level.

```
user@host# set routing-instances mgmt_junos description description
```

3. Configure the routing instance for the BMP routing instance.

```
user@host# set routing-options bmp station station-name routing-instance mgmt_junos
```

4. Configure the connection mode.

   - If you configure passive mode, configure the following additional statements:

```
set routing-options bmp station station-name connection-mode passive
set routing-options bmp station station-name local-address ip-address
set routing-options bmp station station-name local-port port-number
set routing-options bmp station station-name station-address ip-address
```

   - If you configure active mode, configure the following additional statements:

```
set routing-options bmp station station-name connection-mode active
set routing-options bmp station station-name station-address ip-address
set routing-options bmp station station-name station-port port-number
```

RELATED DOCUMENTATION

management-instance

Management Interface in a Non-Default Instance

Example: Configuring the BGP Monitoring Protocol

IN THIS SECTION

- Requirements | 1434
- Overview | 1434
- Configuration | 1435
- Verification | 1437

This example shows how to enable the BGP Monitoring Protocol (BMP). The Junos OS implementation of BMP is based on RFC 8671.

Requirements

- Configure the router interfaces.

  NOTE: When an interface is disabled, the BMP that monitors the TCP session, is shut down for 240 seconds (4 minutes). This is an expected behaviour.

- Configure an interior gateway protocol (IGP).
- Configure BGP and routing policies.
- Configure a monitoring station to listen on a particular TCP port.

Overview

IN THIS SECTION

- Topology | 1435

To configure the monitoring station to which BMP data is sent, you must configure both the `station-address` and `station-port` statements. For the station address, you can specify either the IP address or the name of the monitoring station. For *name*, specify the station name. For the station port, specify a TCP port. BMP operates over TCP. The monitoring station is configured to listen on a particular TCP port,

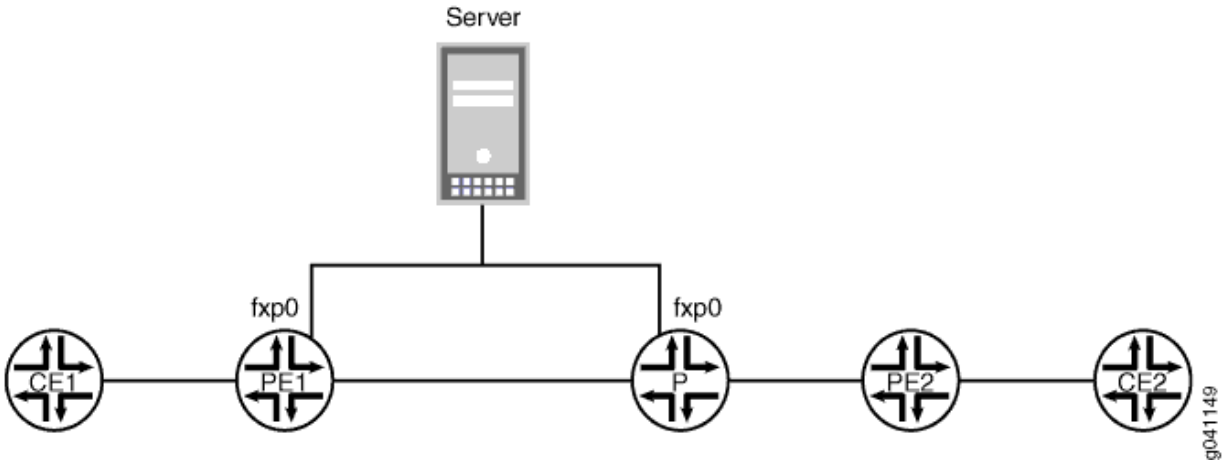
and the router is configured to establish an active connection to that port and to send messages on that TCP connection. You configure BMP in the default routing instance only. However, BMP applies to routes in the default routing instance and to routes in other routing instances.

You can optionally specify how often to send data to the monitoring station. The default is 1 minute. To modify this interval, include the `statistics-timeout` *seconds* statement. For *seconds*, you can specify a value from 15 through 65,535.

Topology

Figure 99 on page 1435 shows a sample topology. In this example, BMP is configured on Router PE1. The server address is 192.168.64.180. The listening TCP port on the server is port 11019.

Figure 99: BMP Topology



Configuration

IN THIS SECTION

- CLI Quick Configuration | **1436**
- Procedure | **1436**
- Results | **1436**

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.

```
set routing-options bmp station-address 192.168.64.180
set routing-options bmp station-port 11019
```

Procedure

Step-by-Step Procedure

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

To configure BMP:

1. Configure the receiving station address.

```
[edit routing-options]
user@PE1# set bmp station-address 192.168.64.180
```

2. Configure the receiving station port.

```
[edit routing-options]
user@PE1# set bmp station-port 11019
```

Results

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

```
user@PE1# show routing-options
bmp {
    station-address 192.168.64.180;
```

```
    station-port 11019;
}
```

Verification

IN THIS SECTION

- Verifying That BMP is Operating | **1437**

Verifying That BMP is Operating

Purpose

Run the `show bgp bmp` command to display a set of statistics and the current BMP session state on the router.

Action

```
user@PE1> show bgp bmp
  BMP station address/port: 192.168.64.180+11019
  BMP session state: DOWN
   Statistics timeout: 15
```

SEE ALSO

Example: Viewing BGP Trace Files on Logical Systems | **1440**

Understanding Trace Operations for BGP Protocol Traffic

You can trace various BGP protocol traffic to help you debug BGP protocol issues. To trace BGP protocol traffic, include the `traceoptions` statement at the `[edit protocols bgp]` hierarchy level. For routing instances,

include the traceoptions statement at the [edit routing-instances *routing-instance-name* protocols bgp] hierarchy level.

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

You can specify the following BGP protocol-specific trace options using the flag statement:

- 4byte-as—4-byte AS events.
- bfd—BFD protocol events.
- damping—Damping operations.
- graceful-restart—Graceful restart events.
- keepalive—BGP keepalive messages.
- nsr-synchronization—Nonstop active routing synchronization events.
- open—BGP open packets. These packets are sent between peers when they are establishing a connection.
- packets—All BGP protocol packets.
- refresh—BGP refresh packets.
- update—BGP update packets. These packets provide routing updates to BGP systems.

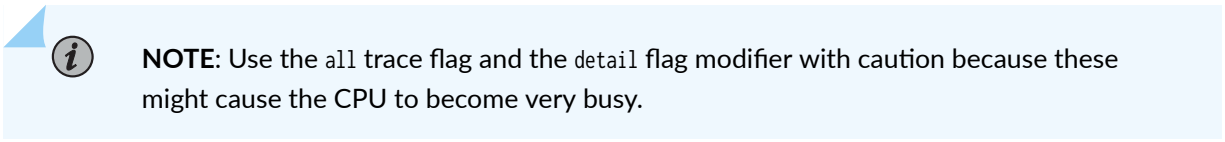
Global tracing options are inherited from the configuration set by the traceoptions statement at the [edit routing-options] hierarchy level. You can override the following global trace options for the BGP protocol using the traceoptions flag statement included at the [edit protocols bgp] hierarchy level:

- all—All tracing operations
- general—All normal operations and routing table changes (a combination of the normal and route trace operations)
- normal—Normal events
- policy—Policy processing
- route—Routing information
- state—State transitions

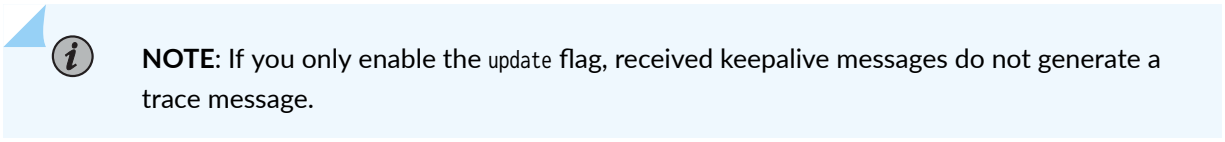
- `task`—Routing protocol task processing
- `timer`—Routing protocol timer processing

You can optionally specify one or more of the following flag modifiers:

- `detail`—Detailed trace information.
- `filter`—Filter trace information. Applies only to `route` and `damping` tracing flags.
- `receive`—Packets being received.
- `send`—Packets being transmitted.



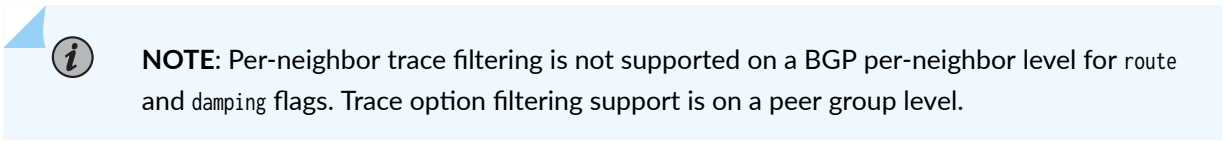
NOTE: Use the `all` trace flag and the `detail` flag modifier with caution because these might cause the CPU to become very busy.



NOTE: If you only enable the `update` flag, received keepalive messages do not generate a trace message.

You can filter trace statements and display only the statement information that passes through the filter by specifying the `filter` flag modifier. The `filter` modifier is only supported for the `route` and `damping` tracing flags.

The `match-on` statement specifies filter matches based on prefixes. It is used to match on route filters.



NOTE: Per-neighbor trace filtering is not supported on a BGP per-neighbor level for `route` and `damping` flags. Trace option filtering support is on a peer group level.

SEE ALSO

traceoptions

Example: Viewing BGP Trace Files on Logical Systems

IN THIS SECTION

- Requirements | 1440
- Overview | 1440
- Configuration | 1441
- Verification | 1447

This example shows how to list and view files that are stored on a logical system.

Requirements

- You must have the **view** privilege for the logical system.
- Configure a network, such as the BGP network shown in "Example: Configuring Internal BGP Peering Sessions on Logical Systems" on page 80.

Overview

Logical systems have their individual directory structure created in the **/var/logical-systems/*logical-system-name*** directory. It contains the following subdirectories:

- **/config**—Contains the active configuration specific to the logical system.
- **/log**—Contains system log and tracing files specific to the logical system.

  To maintain backward compatibility for the log files with previous versions of Junos OS, a symbolic link (symlink) from the **/var/logs/*logical-system-name*** directory to the **/var/logical-systems/*logical-system-name*** directory is created when a logical system is configured.
- **/tmp**—Contains temporary files specific to the logical system.

The file system for each logical system enables logical system users to view trace logs and modify logical system files. Logical system administrators have full access to view and modify all files specific to the logical system.

Logical system users and administrators can save and load configuration files at the logical-system level using the **save** and **load** configuration mode commands. In addition, they can also issue the **show log**, **monitor**, and **file** operational mode commands at the logical-system level.

This example shows how to configure and view a BGP trace file on a logical system. The steps can be adapted to apply to trace operations for any Junos OS hierarchy level that supports trace operations.

TIP: To view a list of hierarchy levels that support tracing operations, enter the **help apropos traceoptions** command in configuration mode.

Configuration

IN THIS SECTION

- CLI Quick Configuration | **1441**
- Configuring Trace Operations | **1441**
- Viewing the Trace File | **1442**
- Deactivating and Reactivating Trace Logging | **1445**
- Results | **1446**

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.

```
set logical-systems A protocols bgp group internal-peers traceoptions file bgp-log
set logical-systems A protocols bgp group internal-peers traceoptions file size 10k
set logical-systems A protocols bgp group internal-peers traceoptions file files 2
set logical-systems A protocols bgp group internal-peers traceoptions flag update detail
```

Configuring Trace Operations

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 the trace operations:

1. Configure trace operations on the logical system.

```
[edit logical-systems A protocols bgp group internal-peers]
user@host# set traceoptions file bgp-log
user@host# set traceoptions file size 10k
user@host# set traceoptions file files 2
user@host# set traceoptions flag update detail
```

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

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

Viewing the Trace File

Step-by-Step Procedure

To view the trace file:

1. In operational mode on the main router, list the directories on the logical system.

```
user@host> file list /var/logical-systems/A
/var/logical-systems/A:
config/
log/
tmp/
```

2. In operational mode on the main router, list the log files on the logical system.

```
user@host> file list /var/logical-systems/A/log/
/var/logical-systems/A/log:
bgp-log
```

3. View the contents of the **bgp-log** file.

```
user@host> file show /var/logical-systems/A/log/bgp-log
Aug 10 17:12:01 trace_on: Tracing to "/var/log/A/bgp-log" started
Aug 10 17:14:22.826182 bgp_peer_mgmt_clear:5829: NOTIFICATION sent to 192.163.6.4 (Internal
```

```
AS 17): code 6 (Cease) subcode 4 (Administratively Reset), Reason: Management session
cleared BGP neighbor
Aug 10 17:14:22.826445 bgp_send: sending 21 bytes to 192.163.6.4 (Internal AS 17)
Aug 10 17:14:22.826499
Aug 10 17:14:22.826499 BGP SEND 192.168.6.5+64965 -> 192.163.6.4+179
Aug 10 17:14:22.826559 BGP SEND message type 3 (Notification) length 21
Aug 10 17:14:22.826598 BGP SEND Notification code 6 (Cease) subcode 4 (Administratively
Reset)
Aug 10 17:14:22.831756 bgp_peer_mgmt_clear:5829: NOTIFICATION sent to 192.168.40.4
(Internal AS 17): code 6 (Cease) subcode 4 (Administratively Reset), Reason: Management
session cleared BGP neighbor
Aug 10 17:14:22.831851 bgp_send: sending 21 bytes to 192.168.40.4 (Internal AS 17)
Aug 10 17:14:22.831901
Aug 10 17:14:22.831901 BGP SEND 192.168.6.5+53889 -> 192.168.40.4+179
Aug 10 17:14:22.831959 BGP SEND message type 3 (Notification) length 21
Aug 10 17:14:22.831999 BGP SEND Notification code 6 (Cease) subcode 4 (Administratively
Reset)
...
```

4. Filter the output of the log file.

```
user@host> file show /var/logical-systems/A/log/bgp-log | match "flags 0x40"
Aug 10 17:14:54.867460 BGP SEND flags 0x40 code Origin(1): IGP
Aug 10 17:14:54.867595 BGP SEND flags 0x40 code ASPath(2) length 0: <null>
Aug 10 17:14:54.867650 BGP SEND flags 0x40 code NextHop(3): 192.168.6.5
Aug 10 17:14:54.867692 BGP SEND flags 0x40 code LocalPref(5): 100
Aug 10 17:14:54.884529 BGP RECV flags 0x40 code Origin(1): IGP
Aug 10 17:14:54.884581 BGP RECV flags 0x40 code ASPath(2) length 0: <null>
Aug 10 17:14:54.884628 BGP RECV flags 0x40 code NextHop(3): 192.163.6.4
Aug 10 17:14:54.884667 BGP RECV flags 0x40 code LocalPref(5): 100
Aug 10 17:14:54.911377 BGP RECV flags 0x40 code Origin(1): IGP
Aug 10 17:14:54.911422 BGP RECV flags 0x40 code ASPath(2) length 0: <null>
Aug 10 17:14:54.911466 BGP RECV flags 0x40 code NextHop(3): 192.168.40.4
Aug 10 17:14:54.911507 BGP RECV flags 0x40 code LocalPref(5): 100
Aug 10 17:14:54.916008 BGP SEND flags 0x40 code Origin(1): IGP
Aug 10 17:14:54.916054 BGP SEND flags 0x40 code ASPath(2) length 0: <null>
Aug 10 17:14:54.916100 BGP SEND flags 0x40 code NextHop(3): 192.168.6.5
Aug 10 17:14:54.916143 BGP SEND flags 0x40 code LocalPref(5): 100
Aug 10 17:14:54.920304 BGP RECV flags 0x40 code Origin(1): IGP
Aug 10 17:14:54.920348 BGP RECV flags 0x40 code ASPath(2) length 0: <null>
```

```
Aug 10 17:14:54.920393 BGP RECV flags 0x40 code NextHop(3): 10.0.0.10
Aug 10 17:14:54.920434 BGP RECV flags 0x40 code LocalPref(5): 100
```

5. View the tracing operations in real time.

```
user@host> clear bgp neighbor logical-system A
Cleared 2 connections
```

CAUTION: Clearing the BGP neighbor table is disruptive in a production environment.

6. Run the **monitor start** command with an optional **match** condition.

```
user@host> monitor start A/bgp-log | match 0.0.0.0/0
Aug 10 19:21:40.773467 BGP RECV         0.0.0.0/0
Aug 10 19:21:40.773685 bgp_rcv_nlri: 0.0.0.0/0
Aug 10 19:21:40.773778 bgp_rcv_nlri: 0.0.0.0/0 belongs to meshgroup
Aug 10 19:21:40.773832 bgp_rcv_nlri: 0.0.0.0/0 qualified bnp->ribact 0x0 l2afcb 0x0
```

7. Pause the **monitor** command by pressing Esc-Q.
   To unpause the output, press Esc-Q again.

8. Halt the **monitor** command by pressing Enter and typing **monitor stop**.

```
[Enter]
user@host> monitor stop
```

9. When you are finished troubleshooting, consider deactivating trace logging to avoid any unnecessary impact to system resources.

```
[edit protocols bgp group internal-peers]
user@host:A# deactivate traceoptions
user@host:A# commit
```

When configuration is deactivated, it appears in the configuration with the **inactive** tag.To reactivate trace operations, use the **activate** configuration-mode statement.

```
[edit protocols bgp group internal-peers]
user@host:A# show

type internal;
inactive: traceoptions {
    file bgp-log size 10k files 2;
    flag update detail;
    flag all;
}
local-address 192.168.6.5;
export send-direct;
neighbor 192.163.6.4;
neighbor 192.168.40.4;
```

10. To reactivate trace operations, use the **activate** configuration-mode statement.

```
[edit protocols bgp group internal-peers]
user@host:A# activate traceoptions
user@host:A# commit
```

Deactivating and Reactivating Trace Logging

Step-by-Step Procedure

To deactivate and reactivate the trace file:

1. When you are finished troubleshooting, consider deactivating trace logging to avoid an unnecessary impact to system resources.

```
[edit protocols bgp group internal-peers]
user@host:A# deactivate traceoptions
user@host:A# commit
```

When configuration is deactivated, the statement appears in the configuration with the **inactive** tag.

```
[edit protocols bgp group internal-peers]
user@host:A# show

type internal;
inactive: traceoptions {
    file bgp-log size 10k files 2;
    flag update detail;
    flag all;
}
local-address 192.168.6.5;
export send-direct;
neighbor 192.163.6.4;
neighbor 192.168.40.4;
```

2. To reactivate logging, use the **activate** configuration-mode statement.

```
[edit protocols bgp group internal-peers]
user@host:A# activate traceoptions
user@host:A# commit
```

Results

From configuration mode, confirm your configuration by entering the **show logical-systems A protocols bgp group internal-peers** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show logical-systems A protocols bgp group internal-peers
traceoptions {
    file bgp-log size 10k files 2;
    flag update detail;
}
```

Verification

IN THIS SECTION

- Verifying That the Trace Log File Is Operating | 1447

Confirm that the configuration is working properly.

Verifying That the Trace Log File Is Operating

Purpose

Make sure that events are being written to the log file.

Action

```
user@host:A> show log bgp-log
Aug 12 11:20:57 trace_on: Tracing to "/var/log/A/bgp-log" started
```

Example: Tracing Global Routing Protocol Operations

IN THIS SECTION

- Requirements | 1448
- Overview | 1448
- Configuration | 1449
- Verification | 1453

This example shows how to list and view files that are created when you enable global routing trace operations.

Requirements

You must have the **view** privilege.

Overview

To configure global routing protocol tracing, include the `traceoptions` statement at the `[edit routing-options]` hierarchy level:

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

The flags in a `traceoptions flag` statement are identifiers. When you use the `set` command to configure a flag, any flags that might already be set are not modified. In the following example, setting the **timer** tracing flag has no effect on the already configured **task** flag. Use the `delete` command to delete a particular flag.

```
[edit routing-options traceoptions]
user@host# show
flag task;
user@host# set traceoptions flag timer
user@host# show
flag task;
flag timer;
user@host# delete traceoptions flag task
user@host# show
flag timer;
```

This example shows how to configure and view a trace file that tracks changes in the routing table. The steps can be adapted to apply to trace operations for any Junos OS hierarchy level that supports trace operations.

TIP: To view a list of hierarchy levels that support tracing operations, enter the `help apropos traceoptions` command in configuration mode.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 1449
- Configuring Trace Operations | 1449
- Viewing the Trace File | 1450
- Results | 1453

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.

```
set routing-options traceoptions file routing-table-changes
set routing-options traceoptions file size 10m
set routing-options traceoptions file files 10
set routing-options traceoptions flag route
set routing-options static route 1.1.1.2/32 next-hop 10.0.45.6
```

Configuring Trace Operations

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 Junos OS CLI User Guide.

To configure the trace operations:

1. Configure trace operations.

```
[edit routing-options traceoptions]
user@host# set file routing-table-changes
user@host# set file size 10m
```

```
user@host# set file files 10
user@host# set flag route
```

2. Configure a static route to cause a change in the routing table.

```
[edit routing-options static]
user@host# set route 1.1.1.2/32 next-hop 10.0.45.6
```

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

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

Viewing the Trace File

Step-by-Step Procedure

To view the trace file:

1. In operational mode, list the log files on the system.

```
user@host> file list /var/log
/var/log:
...
routing-table-changes
...
```

2. View the contents of the **routing-table-changes** file.

```
user@host> file show /var/log/routing-table-changes
Dec 15 11:09:29 trace_on: Tracing to "/var/log/routing-table-changes" started
Dec 15 11:09:29.496507
Dec 15 11:09:29.496507 Tracing flags enabled: route
Dec 15 11:09:29.496507
Dec 15 11:09:29.533203 inet_routerid_notify: Router ID: 192.168.4.1
Dec 15 11:09:29.533334 inet_routerid_notify: No Router ID assigned
Dec 15 11:09:29.533381 inet_routerid_notify: No Router ID assigned
Dec 15 11:09:29.533420 inet_routerid_notify: No Router ID assigned
```

```
Dec 15 11:09:29.534915 inet_routerid_notify: Router ID: 192.168.4.1
Dec 15 11:09:29.542934 inet_routerid_notify: No Router ID assigned
Dec 15 11:09:29.549253 inet_routerid_notify: No Router ID assigned
Dec 15 11:09:29.556878 inet_routerid_notify: No Router ID assigned
Dec 15 11:09:29.582990 rt_static_reinit: examined 3 static nexthops, 0 unreferenced
Dec 15 11:09:29.589920
Dec 15 11:09:29.589920 task_reconfigure reinitializing done
...
```

3. Filter the output of the log file.

```
user@host> file show /var/log/routing-table-changes | match 1.1.1.2
Dec 15 11:15:30.780314 ADD      1.1.1.2/32          nhid 0 gw 10.0.45.6       Static   pref
5/0 metric  at-0/2/0.0 <ctive Int Ext>
Dec 15 11:15:30.782276 KRT Request: send len 216 v104 seq 0 ADD route/user af 2 table 0 infot
0 addr 1.1.1.2 nhop-type unicast nhindex 663
```

4. View the tracing operations in real time by running the `monitor start` command with an optional **match** condition.

```
user@host> monitor start routing-table-changes | match 1.1.1.2
Aug 10 19:21:40.773467 BGP RECV         0.0.0.0/0
Aug 10 19:21:40.773685 bgp_rcv_nlri: 0.0.0.0/0
Aug 10 19:21:40.773778 bgp_rcv_nlri: 0.0.0.0/0 belongs to meshgroup
Aug 10 19:21:40.773832 bgp_rcv_nlri: 0.0.0.0/0 qualified bnp->ribact 0x0 l2afcb 0x0
```

5. Deactivate the static route.

```
user@host# deactivate routing-options static route 1.1.1.2/32
user@host# commit
```

```
*** routing-table-changes ***
Dec 15 11:42:59.355557 CHANGE   1.1.1.2/32          nhid 663 gw 10.0.45.6       Static   pref
5/0 metric  at-0/2/0.0 <Delete Int Ext>
Dec 15 11:42:59.426887 KRT Request: send len 216 v104 seq 0 DELETE route/user af 2 table 0
infot 0 addr 1.1.1.2 nhop-type discard filtidx 0
Dec 15 11:42:59.427366 RELEASE  1.1.1.2/32          nhid 663 gw 10.0.45.6       Static   pref
5/0 metric  at-0/2/0.0 <Release Delete Int Ext>
```

6. Halt the `monitor` command by pressing Enter and typing **monitor stop**.

```
[Enter]
user@host> monitor stop
```

7. When you are finished troubleshooting, consider deactivating trace logging to avoid any unnecessary impact to system resources.

   When configuration is deactivated, it appears in the configuration with the **inactive** tag.

```
[edit routing-options]
user@host# deactivate traceoptions
user@host# commit
```

```
[edit routing-options]
user@host# show

inactive: traceoptions {
    file routing-table-changes size 10m files 10;
    flag route;
}
static {
    inactive: route 1.1.1.2/32 next-hop 10.0.45.6;
}
```

8. To reactivate trace operations, use the **activate** configuration-mode statement.

```
[edit routing-options]
user@host# activate traceoptions
user@host# commit
```

Results

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

```
user@host# show routing-options
traceoptions {
    file routing-table-changes size 10m files 10;
    flag route;
}
static {
    route 1.1.1.2/32 next-hop 10.0.45.6;
}
```

Verification

IN THIS SECTION

- Verifying That the Trace Log File Is Operating | **1453**

Confirm that the configuration is working properly.

Verifying That the Trace Log File Is Operating

Purpose

Make sure that events are being written to the log file.

Action

```
user@host> show log routing-table-changes
Dec 15 11:09:29 trace_on: Tracing to "/var/log/routing-table-changes" started
```

Tracing BMP Operations

You can trace BMP operations for all BMP stations by configuring the `traceoptions` statement at the `[edit routing-options bmp]` hierarchy level or for specific BMP stations at the `[edit routing-options bmp station` *`station-name`*`]` hierarchy level.

To trace BMP operations, complete the following steps:

1. Configure the `traceoptions` statement:

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

2. Specify the name of the file to receive the output of the tracing operation using the `file` option. Enclose the name within quotation marks. All files are placed in the directory **/var/log**. We recommend that you place BMP tracing output in the file **bmp-log**.
3. (Optional) Specify the maximum number of trace files using the `files` option. When a trace file named ***trace-file*.0** reaches its maximum size, it is renamed ***trace-file*.0**, then ***trace-file*.1**, and so on, until the maximum number of trace files is reached. Then, the oldest trace file is overwritten. If you specify a maximum number of files, you must also specify a maximum file size with the `size` option.
4. (Optional) Specify the maximum size of each trace file using the `size` option in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When the ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then, the oldest trace file is overwritten. If you specify a maximum file size, you also must specify a maximum number of trace files with the `files` option.
5. (Optional) You can specify that the log files are either `world-readable` (accessible to all users on the device) or `no-world-readable` (not accessible to all users on the device).
6. You can specify the following BMP-specific trace options using the `flag` statement:
   - `all`—Trace all BMP monitoring operations.
   - `down`—Down messages.
   - `error`—Error conditions.
   - `event`—Major events, station establishment, errors, and events.
   - `general`—General events.
   - `normal`—Normal events.

- packets—All messages.
- policy—Policy processing.
- route—Routing information.
- route-monitoring—Route monitoring messages.
- state—State transitions.
- statistics—Statistics messages.
- task—Routing protocol task processing.
- timer—Routing protocol timer processing.
- up—Up messages.
- write—Writing of messages.

You can optionally specify one or more of the following flag modifiers:

- detail—Provide detailed trace information.
- disable—Disable the tracing flag.
- receive—Trace the packets being received.
- send—Trace the packets being transmitted.

NOTE: Use the all trace flag and the detail flag modifier with caution due to the increased computer processing power required.

SEE ALSO

Configuring BGP Monitoring Protocol Version 3 | **1430**

Change History Table

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

Release	Description
18.3R1	Starting in Junos OS Release 18.3R1, you can specify which routing instance you want the BGP Monitoring Protocol (BMP) to use.

Troubleshooting Network Issues

IN THIS SECTION

- Working with Problems on Your Network | 1456
- Isolating a Broken Network Connection | 1457
- Identifying the Symptoms of a Broken Network Connection | 1459
- Isolating the Causes of a Network Problem | 1461
- Taking Appropriate Action for Resolving the Network Problem | 1462
- Evaluating the Solution to Check Whether the Network Problem Is Resolved | 1464
- Checklist for Tracking Error Conditions | 1466
- Configure Routing Protocol Process Tracing | 1468
- Configure Routing Protocol Tracing for a Specific Routing Protocol | 1471
- Monitor Trace File Messages Written in Near-Real Time | 1474
- Stop Trace File Monitoring | 1475

Working with Problems on Your Network

IN THIS SECTION

- Problem | 1456
- Solution | 1457

Problem

Description

This checklist provides links to troubleshooting basics, an example network, and includes a summary of the commands you might use to diagnose problems with the router and network.

Solution

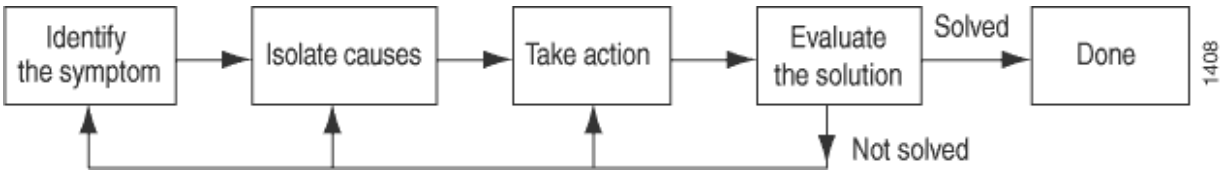
Table 13: Checklist for Working with Problems on Your Network

Tasks	Command or Action
"Isolating a Broken Network Connection" on page 1457	
1. "Identifying the Symptoms of a Broken Network Connection" on page 1459	**ping (*ip-address* | *hostname*) show route (*ip-address* | *hostname*) traceroute (*ip-address* | *hostname*)**
1. "Isolating the Causes of a Network Problem" on page 1461	`show < configuration | interfaces | protocols | route >`
1. "Taking Appropriate Action for Resolving the Network Problem" on page 1456	`[edit] delete routing options static route` *`destination-prefix`* **commit and-quit show route *destination-prefix***
1. "Evaluating the Solution to Check Whether the Network Problem Is Resolved" on page 1464	`show route (`*`ip-address`*` | `*`hostname`*`) ping (`*`ip-address`*` | `*`hostname`*`)` **count 3 traceroute (*ip-address* | *hostname*)**

Isolating a Broken Network Connection

By applying the standard four-step process illustrated in Figure 100 on page 1457, you can isolate a failed node in the network. Note that the functionality described in this section is not supported in versions 15.1X49, 15.1X49-D30, or 15.1X49-D40.

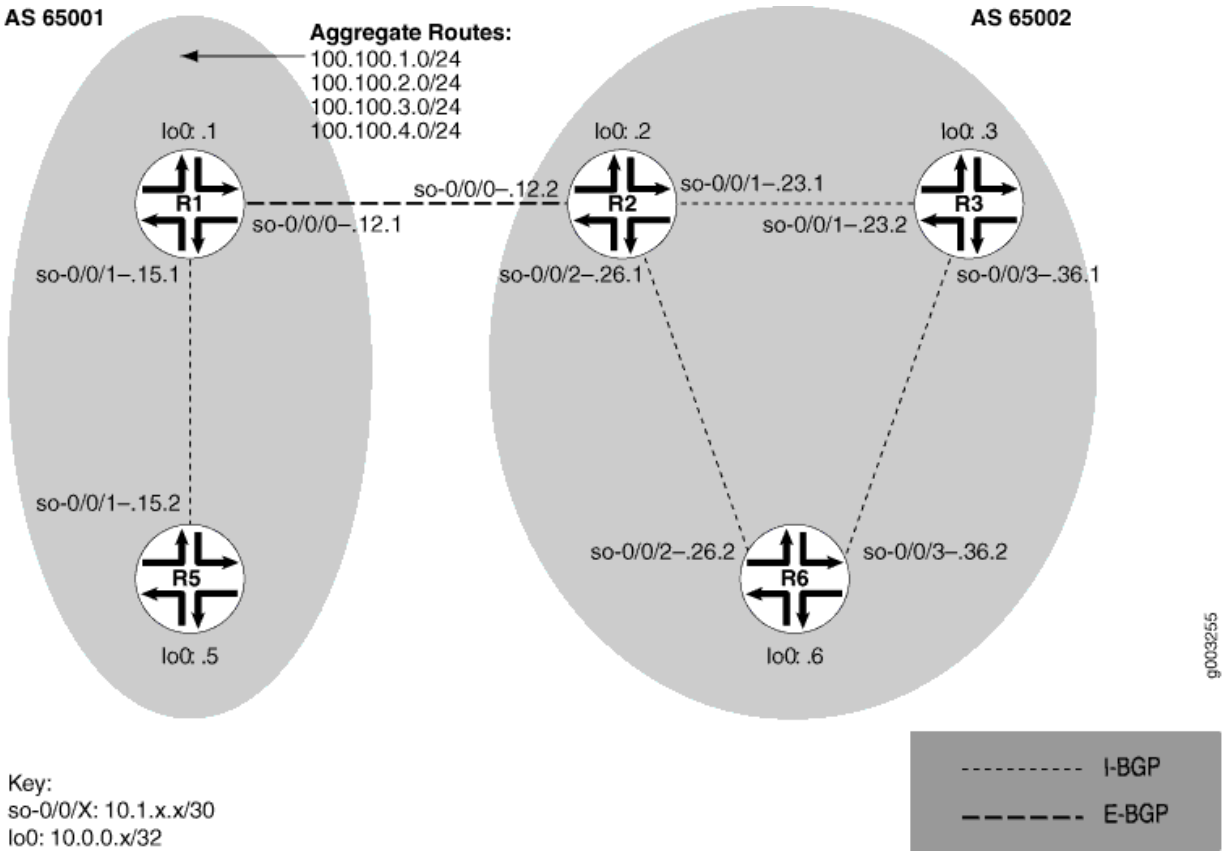
Figure 100: Process for Diagnosing Problems in Your Network



Before you embark on the four-step process, however, it is important that you are prepared for the inevitable problems that occur on all networks. While you might find a solution to a problem by simply trying a variety of actions, you can reach an appropriate solution more quickly if you are systematic in your approach to the maintenance and monitoring of your network. To prepare for problems on your network, understand how the network functions under normal conditions, have records of baseline network activity, and carefully observe the behavior of your network during a problem situation.

Figure 101 on page 1458 shows the network topology used in this topic to illustrate the process of diagnosing problems in a network.

Figure 101: Network with a Problem



The network in Figure 101 on page 1458 consists of two autonomous systems (ASs). AS 65001 includes two routers, and AS 65002 includes three routers. The border router (`R1`) in AS 65001 announces aggregated prefixes `100.100/24` to the AS 65002 network. The problem in this network is that `R6` does not have access to `R5` because of a loop between `R2` and `R6`.

To isolate a failed connection in your network, follow the steps in these topics:

- "Isolating the Causes of a Network Problem" on page 1461

- "Taking Appropriate Action for Resolving the Network Problem" on page 1456
- "Taking Appropriate Action for Resolving the Network Problem" on page 1456
- "Evaluating the Solution to Check Whether the Network Problem Is Resolved" on page 1464

Identifying the Symptoms of a Broken Network Connection

IN THIS SECTION

- Problem | 1459
- Solution | 1459

Problem

Description

The symptoms of a problem in your network are usually quite obvious, such as the failure to reach a remote host.

Solution

To identify the symptoms of a problem on your network, start at one end of your network and follow the routes to the other end, entering all or one of the following Junos OS command-line interfaces (CLI) operational mode commands:

```
user@host> ping (ip-address | host-name)
user@host> show route (ip-address | host-name)
user@host> traceroute (ip-address | host-name)
```

Sample Output

```
user@R6> ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5): 56 data bytes
36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
```

```
 4  5  00 0054 e2db   0 0000  01  01 a8c6 10.1.26.2  10.0.0.5

36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 e2de   0 0000  01  01 a8c3 10.1.26.2  10.0.0.5

36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 e2e2   0 0000  01  01 a8bf 10.1.26.2  10.0.0.5

^C
--- 10.0.0.5 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss

user@R6> show route 10.0.0.5

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[IS-IS/165] 00:02:39, metric 10
                     > to 10.1.26.1  via so-0/0/2.0

user@R6> traceroute 10.0.0.5
traceroute to 10.0.0.5 (10.0.0.5), 30 hops max, 40 byte packets
 1  10.1.26.1 (10.1.26.1)  0.649 ms  0.521 ms  0.490 ms
 2  10.1.26.2 (10.1.26.2)  0.521 ms  0.537 ms  0.507 ms
 3  10.1.26.1 (10.1.26.1)  0.523 ms  0.536 ms  0.514 ms
 4  10.1.26.2 (10.1.26.2)  0.528 ms  0.551 ms  0.523 ms
 5  10.1.26.1 (10.1.26.1)  0.531 ms  0.550 ms  0.524 ms
```

Meaning

The sample output shows an unsuccessful `ping` command in which the packets are being rejected because the time to live is exceeded. The output for the `show route` command shows the interface (`10.1.26.1`) that you can examine further for possible problems. The `traceroute` command shows the loop between `10.1.26.1` (R2) and `10.1.26.2` (R6), as indicated by the continuous repetition of the two interface addresses.

Isolating the Causes of a Network Problem

IN THIS SECTION

- Problem | 1461
- Solution | 1461

Problem

Description

A particular symptom can be the result of one or more causes. Narrow down the focus of your search to find each individual cause of the unwanted behavior.

Solution

To isolate the cause of a particular problem, enter one or all of the following Junos OS CLI operational mode command:

```
user@host> show < configuration | bgp | interfaces | isis | ospf | route
>
```

Your particular problem may require the use of more than just the commands listed above. See the appropriate command reference for a more exhaustive list of commonly used operational mode commands.

Sample Output

```
user@R6> show interfaces terse
Interface               Admin Link Proto Local                 Remote
so-0/0/0                up    up
so-0/0/0.0              up    up   inet  10.1.56.2/30
                                   iso
so-0/0/2                up    up
so-0/0/2.0              up    up   inet  10.1.26.2/30
                                   iso
so-0/0/3                up    up
```

```
so-0/0/3.0              up    up   inet  10.1.36.2/30
                                   iso
[...Output truncated...]
```

The following sample output is from R2:

```
user@R2> show route 10.0.0.5

inet.0: 22 destinations, 25 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32         *[Static/5] 00:16:21
                    > to 10.1.26.2 via so-0/0/2.0
                    [BGP/170] 3d 20:23:35, MED 5, localpref 100
                      AS path: 65001 I
                    > to 10.1.12.1 via so-0/0/0.0
```

Meaning

The sample output shows that all interfaces on R6 are up. The output from R2 shows that a static route [Static/5] configured on R2 points to R6 (10.1.26.2) and is the preferred route to R5 because of its low preference value. However, the route is looping from R2 to R6, as indicated by the missing reference to R5 (10.1.15.2).

Taking Appropriate Action for Resolving the Network Problem

IN THIS SECTION

- Problem | 1463
- Solution | 1463

Problem

Description

The appropriate action depends on the type of problem you have isolated. In this example, a static route configured on R2 is deleted from the [routing-options] hierarchy level. Other appropriate actions might include the following:

Solution

- Check the local router's configuration and edit it if appropriate.
- Troubleshoot the intermediate router.
- Check the remote host configuration and edit it if appropriate.
- Troubleshoot routing protocols.
- Identify additional possible causes.

To resolve the problem in this example, enter the following Junos OS CLI commands:

```
[edit]
                          user@R2# delete routing-options static route destination-
prefix
user@R2# commit and-quit
user@R2# show route destination-prefix
```

Sample Output

```
[edit]
user@R2# delete routing-options static route 10.0.0.5/32

[edit]
user@R2# commit and-quit
commit complete
Exiting configuration mode

user@R2> show route 10.0.0.5

inet.0: 22 destinations, 24 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
10.0.0.5/32        *[BGP/170] 3d 20:26:17, MED 5, localpref 100
                      AS path: 65001 I
                    > to 10.1.12.1 via so-0/0/0.0
```

Meaning

The sample output shows the static route deleted from the [routing-options] hierarchy and the new configuration committed. The output for the show route command now shows the BGP route as the preferred route, as indicated by the asterisk (*).

Evaluating the Solution to Check Whether the Network Problem Is Resolved

IN THIS SECTION

- Problem | 1464
- Solution | 1465

Problem

Description

If the problem is solved, you are finished. If the problem remains or a new problem is identified, start the process over again.

You can address possible causes in any order. In relation to the network in "Isolating a Broken Network Connection" on page 1457, we chose to work from the local router toward the remote router, but you might start at a different point, particularly if you have reason to believe that the problem is related to a known issue, such as a recent change in configuration.

Solution

To evaluate the solution, enter the following Junos OS CLI commands:

```
                                    user@host> show route (ip-address   |host-name)
user@host> ping (ip-address  | host-name)
user@host> traceroute (ip-address  | host-name)
```

Sample Output

```
user@R6> show route 10.0.0.5

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32         *[BGP/170]  00:01:35, MED 5, localpref 100, from 10.0.0.2
                      AS path: 65001 I
                    > to 10.1.26.1 via so-0/0/2.0

user@R6> ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5): 56 data bytes
64 bytes from 10.0.0.5: icmp_seq=0 ttl=253 time=0.866 ms
64 bytes from 10.0.0.5: icmp_seq=1 ttl=253 time=0.837 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=253 time=0.796 ms
^C
--- 10.0.0.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.796/0.833/0.866/0.029 ms

user@R6> traceroute 10.0.0.5
traceroute to 10.0.0.5 (10.0.0.5), 30 hops max, 40 byte packets
 1  10.1.26.1 (10.1.26.1)  0.629 ms  0.538 ms  0.497 ms
 2  10.1.12.1 (10.1.12.1)  0.534 ms  0.538 ms  0.510 ms
 3  10.0.0.5 (10.0.0.5)  0.776 ms  0.705 ms  0.672 ms
```

Meaning

The sample output shows that there is now a connection between R6 and R5. The `show route` command shows that the BGP route to R5 is preferred, as indicated by the asterisk (*). The `ping` command is successful and the `traceroute` command shows that the path from R6 to R5 is through R2 (10.1.26.1), and then through R1 (10.1.12.1).

Checklist for Tracking Error Conditions

IN THIS SECTION

- Problem | 1466
- Solution | 1466

Problem

Description

Table 14 on page 1466 provides links and commands for configuring routing protocol daemon tracing, Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS) protocol, and Open Shortest Path First (OSPF) protocol tracing to diagnose error conditions.

Solution

Table 14: Checklist for Tracking Error Conditions

Tasks	Command or Action
Configure Routing Protocol Process Tracing	
1. "Configure Routing Protocol Process Tracing" on page 1468	`[edit] edit routing-options traceoptions` *filename* `size` *size* **files** *number* **show co** **log** *filename*
1. "Configure Routing Protocol Tracing for a Specific Routing Protocol" on page 1471	`[edit] edit protocol` *protocol-name* **trace** *filename* **size** *size* **files** *number* **show com** **log** *filename*
1. "Monitor Trace File Messages Written in Near-Real Time" on page 1474	**monitor start** *filename*
1. "Stop Trace File Monitoring " on page 1475	**monitor stop** *filename*

Table 14: Checklist for Tracking Error Conditions *(Continued)*

Tasks	Command or Action
Configure BGP-Specific Options	
1. Display Detailed BGP Protocol Information	[edit] edit protocol bgp traceoptions se detail show commit run show log *filename*
1. Display Sent or Received BGP Packets	[edit] edit protocol bgp traceoptions se (send | receive) show commit run show lo
1. Diagnose BGP Session Establishment Problems	[edit] edit protocol bgp set traceoptior detail show commit run show log *filename*
Configure IS-IS-Specific Options	
1. Displaying Detailed IS-IS Protocol Information	[edit] edit protocol isis traceoptions s detail show commit run show log *filename*
1. Displaying Sent or Received IS-IS Protocol Packets	[edit] edit protocols isis traceoptions (send | receive) show commit run show lo
1. Analyzing IS-IS Link-State PDUs in Detail	[edit] edit protocols isis traceoptions detail show commit run show log *filename*
Configure OSPF-Specific Options	
1. Diagnose OSPF Session Establishment Problems	[edit] edit protocols ospf traceoptions detail show commit run show log *filename*
1. Analyze OSPF Link-State Advertisement Packets in Detail	[edit] edit protocols ospf traceoptions update detail show commit run show log

Configure Routing Protocol Process Tracing

IN THIS SECTION

- Action | 1468
- Meaning | 1470

Action

To configure routing protocol process (rpd) tracing, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host# edit routing-options traceoptions
```

2. Configure the file, file size, number, and flags:

```
[edit routing-options traceoptions]
user@host#  set file filename size size file number
[edit routing-options traceoptions]
user@host#  set flag flag
```

For example:

```
[edit routing-options traceoptions]
user@host# set file daemonlog size 10240 files 10
[edit routing-options traceoptions]
user@host# set flag general
```

3. Verify the configuration:

```
user@host# show
```

For example:

```
[edit routing-options traceoptions]
user@host# show
file daemonlog size 10k files 10;
flag general;
```

4. Commit the configuration:

```
user@host# commit
```

NOTE: Some traceoptions flags generate an extensive amount of information. Tracing can also slow down the operation of routing protocols. Delete the traceoptions configuration if you no longer require it.

1. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit routing-options traceoptions]
user@pro4-a# run show log daemonlog
Sep 17 14:17:31 trace_on: Tracing to "/var/log/daemonlog" started
Sep 17 14:17:31 Tracing flags enabled: general
Sep 17 14:17:31 inet_routerid_notify: Router ID: 10.255.245.44
Sep 17 14:17:31 inet_routerid_notify: No Router ID assigned
Sep 17 14:17:31 Initializing LSI globals
Sep 17 14:17:31 LSI initialization complete
Sep 17 14:17:31 Initializing OSPF instances
Sep 17 14:17:31 Reinitializing OSPFv2 instance master
Sep 17 14:17:31 OSPFv2 instance master running
[...Output truncated...]
```

Meaning

Table 15 on page 1470 lists tracing flags and example output for Junos-supported routing protocol daemon tracing.

Table 15: Routing Protocol Daemon Tracing Flags

Tracing Flag	Description	Example Output
all	All operations	Not available.
general	Normal operations and routing table change	Not available.
normal	Normal operations	Not available.
policy	Policy operations and actions	**Nov 29 22:19:58 export: Dest 10.0.0.0 proto Static Nov 29 22:19:58 policy_match_qual_or: Qualifier proto Sense: 0 Nov 29 22:19:58 policy_match_qual_or: Qualifier proto Sense: 0 Nov 29 22:19:58 export: Dest 10.10.10.0 proto IS-IS**
route	Routing table changes	**Nov 29 22:23:59 Nov 29 22:23:59 rtlist_walker_job: rt_list walk for RIB inet.0 started with 42 entries Nov 29 22:23:59 rt_flash_update_callback: flash KRT (inet.0) start Nov 29 22:23:59 rt_flash_update_callback: flash KRT (inet.0) done Nov 29 22:23:59 rtlist_walker_job: rt_list walk for inet.0 ended with 42 entries Nov 29 22:23:59 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 CHANGE route/user af 2 addr 172.16.0.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 ADD route/user af 2 addr 172.17.0.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 ADD route/user af 2 addr 10.149.3.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:24:19 trace_on: Tracing to "/var/log/rpdlog" started Nov 29 22:24:19 KRT Request: send len 68 v14 seq 0 DELETE route/user af 2 addr 10.10.218.0 nhop-type unicast nhop 10.10.10.29 Nov 29 22:24:19 RELEASE 10.10.218.0 255.255.255.0 gw 10.10.10.29,10.10.10.33 BGP pref 170/-101 metric so-1/1/0.0,so-1/1/1.0 <Release Delete Int Ext> as 65401 Nov 29 22:24:19 KRT Request: send len 68 v14 seq 0 DELETE route/user af 2 addr 172.18.0.0 nhop-type unicast nhop 10.10.10.33**
state	State transitions	Not available.

Table 15: Routing Protocol Daemon Tracing Flags *(Continued)*

Tracing Flag	Description	Example Output
task	Interface transactions and processing	**Nov 29 22:50:04 foreground dispatch running job task_collect for task Scheduler Nov 29 22:50:04 task_collect_job: freeing task MGMT_Listen (DELETED) Nov 29 22:50:04 foreground dispatch completed job task_collect for task Scheduler Nov 29 22:50:04 background dispatch running job rt_static_update for task RT Nov 29 22:50:04 task_job_delete: delete background job rt_static_update for task RT Nov 29 22:50:04 background dispatch completed job rt_static_update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 background dispatch returned job Flash update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 task_job_delete: delete background job Flash update for task RT Nov 29 22:50:04 background dispatch completed job Flash update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 task_job_delete: delete background job Flash update for task RT**
timer	Timer usage	**Nov 29 22:52:07 task_timer_hiprio_dispatch: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 1 timer Nov 29 22:52:07 task_timer_hiprio_dispatch: running high priority timer queue Nov 29 22:52:07 task_timer_hiprio_dispatch: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 2 timers**

Configure Routing Protocol Tracing for a Specific Routing Protocol

IN THIS SECTION

- Action | **1471**
- Meaning | **1473**

Action

To configure routing protocol tracing for a specific routing protocol, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
                user@host# edit protocol protocol-name traceoptions
```

2. Configure the file, file size, number, and flags:

```
[edit protocols protocol name traceoptions]
                user@host# set file filename size size files
number
[edit protocols protocol name traceoptions]
user@host# set flag flag
```

   For example:

```
[edit protocols ospf traceoptions]
user@host# set file ospflog size 10240 files 10
[edit protocols ospf traceoptions]
user@host# set flag general
```

3. Verify the configuration:

```
user@host# show
```

   For example:

```
[edit protocols ospf traceoptions]
user@host# show
file ospflog size 10k files 10;
flag general;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit protocols ospf traceoptions]
user@pro4-a# run show log ospflog
Sep 17 14:23:10 trace_on: Tracing to "/var/log/ospflog" started
Sep 17 14:23:10 rt_flash_update_callback: flash OSPF (inet.0) start
Sep 17 14:23:10 OSPF:  multicast address 224.0.0.5/32, route ignored
Sep 17 14:23:10 rt_flash_update_callback: flash OSPF (inet.0) done
Sep 17 14:23:10 CHANGE   10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Delete Int>
Sep 17 14:23:10 CHANGE   10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 ADD      10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 CHANGE   10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Delete Int>
Sep 17 14:23:10 CHANGE   10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 ADD      10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 rt_close: 4/4 routes proto OSPF
[...Output truncated...]
```

Meaning

Table 16 on page 1473 lists standard tracing options that are available globally or that can be applied to specific protocols. You can also configure tracing for a specific BGP peer or peer group. For more information, see the *Junos System Basics Configuration Guide*.

Table 16: Standard Trace Options for Routing Protocols

Tracing Flag	Description
all	All operations

Table 16: Standard Trace Options for Routing Protocols *(Continued)*

Tracing Flag	Description
general	Normal operations and routing table changes
normal	Normal operations
policy	Policy operations and actions
route	Routing table changes
state	State transitions
task	Interface transactions and processing
timer	Timer usage

Monitor Trace File Messages Written in Near-Real Time

IN THIS SECTION

- Purpose | **1474**
- Action | **1475**

Purpose

To monitor messages in near-real time as they are being written to a trace file.

Action

To monitor messages in near-real time as they are being written to a trace file, use the following Junos OS command-line interface (CLI) operational mode command:

```
user@host> monitor start  filename
```

Sample Output

command-name

```
user@host> monitor start isis
user@host>
*** isis ***
Sep 15 18:32:21 Updating LSP isis5.02-00 in database
Sep 15 18:32:21 Updating L2 LSP isis5.02-00 in TED
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Scheduling L2 LSP isis5.02-00 sequence 0xd87 on interface fxp2.3
Sep 15 18:32:21 Updating LSP isis5.00-00 in database
Sep 15 18:32:21 Updating L1 LSP isis5.00-00 in TED
Sep 15 18:32:21 Sending L2 LSP isis5.02-00 on interface fxp2.3
Sep 15 18:32:21     sequence 0xd87, checksum 0xc1c8, lifetime 1200
```

Stop Trace File Monitoring

IN THIS SECTION

- Action | 1476
- Sample Output | 1476

Action

To stop monitoring a trace file in near-real time, use the following Junos OS CLI operational mode command after you have started monitoring:

```
user@host              monitor stop filename
```

Sample Output

```
user@host> monitor start isis
user@host>
*** isis ***
Sep 15 18:32:21 Updating LSP isis5.02-00 in database
Sep 15 18:32:21 Updating L2 LSP isis5.02-00 in TED
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Scheduling L2 LSP isis5.02-00 sequence 0xd87 on interface fxp2.3
Sep 15 18:32:21 Updating LSP isis5.00-00 in database
Sep 15 18:32:21 Updating L1 LSP isis5.00-00 in TED
Sep 15 18:32:21 Sending L2 LSP isis5.02-00 on interface fxp2.3
Sep 15 18:32:21     sequence 0xd87, checksum 0xc1c8, lifetime 1200
monitor stop isis
user@host>
```

Troubleshooting BGP Sessions

IN THIS SECTION

- Checklist for Verifying the BGP Protocol and Peers | **1477**
- Verify BGP Peers | **1479**
- Examine BGP Routes and Route Selection | **1492**
- Checklist for Checking the BGP Layer | **1502**

- Checking the BGP Layer | **1503**
- Display Sent or Received BGP Packets | **1523**
- Understanding Hidden Routes | **1525**
- Examine Routes in the Forwarding Table | **1527**
- Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers | **1528**
- Log BGP State Transition Events | **1533**
- Configure BGP-Specific Options | **1536**
- Configure IS-IS-Specific Options | **1541**
- Configure OSPF-Specific Options | **1550**

Checklist for Verifying the BGP Protocol and Peers

IN THIS SECTION

- Purpose | **1477**
- Action | **1477**

Purpose

Table 17 on page 1478 provides links and commands for verifying whether the Border Gateway Protocol (BGP) is configured correctly on a Juniper Networks router in your network, the internal Border Gateway Protocol (IBGP) and exterior Border Gateway Protocol (EBGP) sessions are properly established, the external routes are advertised and received correctly, and the BGP path selection process is working properly.

Action

Table 17: Checklist for Verifying the BGP Protocol and Peers

Tasks	Command or Action
"Verify BGP Peers" on page 1479	
1. "Verify BGP on an Internal Router" on page 1476	`show configuration`
1. "Verify BGP on a Border Router" on page 1476	`show configuration`
1. "Verify Advertised BGP Routes" on page 1476	`show route advertising-protocol bgp` *`neighbor-address`*
1. "Verify That a Particular BGP Route Is Received on Your Router" on page 1476	`show route receive-protocol bgp` *`neighbor-address`*
"Examine BGP Routes and Route Selection" on page 1492	
1. "Examine the Local Preference Selection" on page 1476	`show route` *`destination-prefix`* `< detail >`
1. "Examine the Multiple Exit Discriminator Route Selection" on page 356	`show route` *`destination-prefix`* `< detail >`
1. "Examine the EBGP over IBGP Selection" on page 1476	`show route` *`destination-prefix`* `< detail >`
1. "Examine the IGP Cost Selection" on page 1476	`show route` *`destination-prefix`* `< detail >`
"Examine Routes in the Forwarding Table" on page 1527	`show route forwarding-table`

Verify BGP Peers

IN THIS SECTION

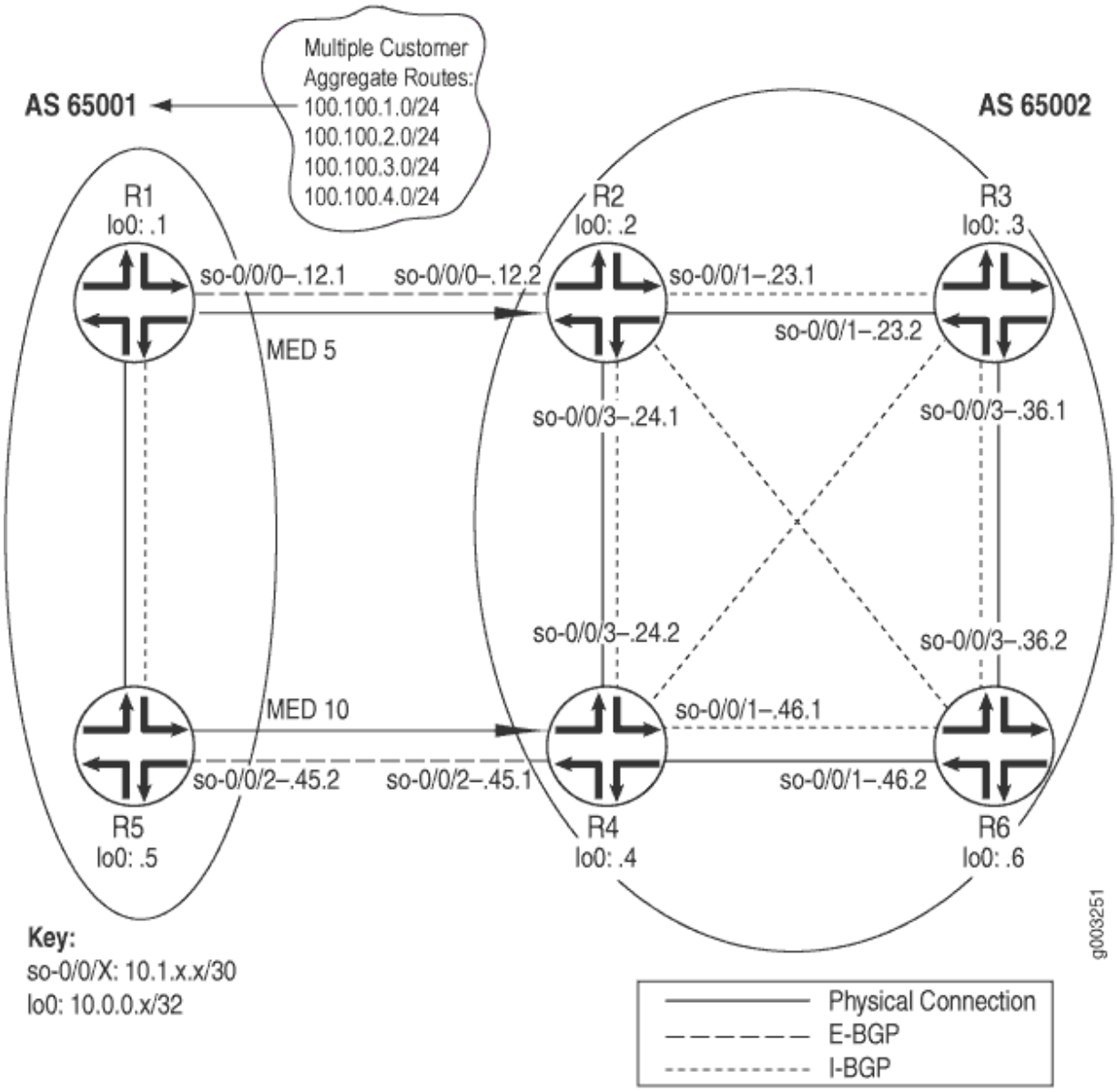
- Verify BGP on an Internal Router | **1481**
- Verify BGP on a Border Router | **1485**
- Verify Advertised BGP Routes | **1490**
- Verify That a Particular BGP Route Is Received on Your Router | **1491**

Purpose

Assuming that all the routers are correctly configured for BGP, you can verify if IBGP and EBGP sessions are properly established, external routes are advertised and received correctly, and the BGP path selection process is working properly.

Figure 102 on page 1480 illustrates an example BGP network topology used in this topic.

Figure 102: BGP Network Topology



The network consists of two directly connected ASs consisting of external and internal peers. The external peers are directly connected through a shared interface and are running EBGP. The internal peers are connected through their loopback (`lo0`) interfaces through IBGP. AS 65001 is running OSPF and AS 65002 is running IS-IS as its underlying IGP. IBGP routers do not have to be directly connected, the underlying IGP allows neighbors to reach one another.

The two routers in AS 65001 each contain one EBGP link to AS 65002 (`R2` and `R4`) over which they announce aggregated prefixes: `100.100.1.0`, `100.100.2.0`, `100.100.3.0`, and `100.100.4.0`. Also, `R1` and `R5` are injecting multiple exit discriminator (MED) values of 5 and 10, respectively, for some routes.

The internal routers in both ASs are using a full mesh IBGP topology. A full mesh is required because the networks are not using confederations or route reflectors, so any routes learned through IBGP are not distributed to other internal neighbors. For example, when R3 learns a route from R2, R3 does not distribute that route to R6 because the route is learned through IBGP, so R6 must have a direct BGP connection to R2 to learn the route.

In a full mesh topology, only the border router receiving external BGP information distributes that information to other routers within its AS. The receiving router does not redistribute that information to other IBGP routers in its own AS.

From the point of view of AS 65002, the following sessions should be up:

- The four routers in AS 65002 should have IBGP sessions established with each other.
- R2 should have an EBGP session established with R1.
- R4 should have an EBGP session established with R5.

To verify BGP peers, follow these steps:

Verify BGP on an Internal Router

IN THIS SECTION

- Purpose | **1481**
- Action | **1481**
- Meaning | **1484**

Purpose

To verify the BGP configuration of an internal router.

Action

To verify the BGP configuration of an internal router, enter the following Junos OS command-line interface (CLI) command:

```
user@host>  show configuration
```

The following sample output is for a BGP configuration on R3:

Sample Output

command-name

```
user@R3> show configuration
[...Output truncated...]
interfaces {
    so-0/0/1 {
        unit 0 {
             family inet {
                address 10.1.23.2/30;
            }
            family iso;
        }
    }
    so-0/0/3 {
        unit 0 {
            family inet {
                address 10.1.36.1/30;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.0.3/32;
            }
            family iso {
                address 49.0002.1000.0000.0003.00;
            }
        }
    }
}
routing-options {
    [...Output truncated...]
    router-id 10.0.0.3;
    autonomous-system 65002;
}
protocols {
    bgp {
        group internal {
```

```
            type internal;
            local-address 10.0.0.3;
            neighbor 10.0.0.2;
            neighbor 10.0.0.4;
            neighbor 10.0.0.6;
        }
    }
    isis {
        level 1 disable;
        interface all {
            level 2 metric 10;
        }
        interface lo0.0;
    }
}

user@R6> show configuration |
[Output truncated...]
interfaces {
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.1.46.2/30;
            }
            family iso;
        }
    }
    so-0/0/3 {
        unit 0 {
            family inet {
                address 10.1.36.2/30;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.0.6/32;
            }
            family iso {
                address 49.0003.1000.0000.0006.00;
            }
```

```
        }
    }
}
routing-options {
    [Output truncated...]
    router-id 10.0.0.6;
    autonomous-system 65002;
}
protocols {
    bgp {
        group internal {
            type internal;
            local-address 10.0.0.6;
            neighbor 10.0.0.2;
            neighbor 10.0.0.3;
            neighbor 10.0.0.4;
        }
    }
    isis {
        level 1 disable;
        interface all {
            level 2 metric 10;
        }
        interface lo0.0;
    }
}
```

Meaning

The sample output shows a basic BGP configuration on routers R3 and R6. The local AS (65002) and one group (internal) are configured on both routers. R3 has three internal peers—10.0.0.2, 10.0.0.4, and 10.0.0.6—included at the [protocols bgp group *group*] hierarchy level. R6 also has three internal peers: 10.0.0.2, 10.0.0.3, and 10.0.0.4. The underlying IGP protocol is Intermediate System-to-Intermediate System (IS-IS), and relevant interfaces are configured to run IS-IS.

Note that in this configuration the router ID is manually configured to avoid any duplicate router ID problems.

Verify BGP on a Border Router

IN THIS SECTION

- Purpose | 1485
- Action | 1485
- Meaning | 1489

Purpose

To verify the BGP configuration of a border router.

Action

To verify the BGP configuration of a border router, enter the following Junos OS CLI operational mode command:

```
user@host>  show configuration
```

Sample Output

command-name

The following sample output is for a BGP configuration on two border routers, R2 and R4 from AS 65002:

```
user@R2> show configuration
[...Output truncated...]
interfaces {
    so-0/0/0 {
        unit 0 {
            family inet {
                address 10.1.12.2/30;
            }
            family iso;
        }
    }
```

```
    so-0/0/1 {
        unit 0 {
             family inet {
                address 10.1.23.1/30;
            }
            family iso;
        }
    }
    so-0/0/3 {
        unit 0 {
             family inet {
                address 10.1.24.1/30;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
             family inet {
                address 10.0.0.2/32;
            }
            family iso {
                address 49.0002.1000.0000.0002.00;
            }
        }
    }
}
routing-options {
[...Output truncated...]
     router-id 10.0.0.2;
    autonomous-system 65002;
}
protocols {
     bgp {
        group internal {
            type internal;
            export next-hop-self;
            neighbor 10.0.0.3;
            neighbor 10.0.0.4;
            neighbor 10.0.0.6;
        }
        group toR1 {
            type external;
```

```
            import import-toR1;
            peer-as 65001;
            neighbor 10.1.12.1;
        }
    }
    isis {
        level 1 disable;
        interface all {
            level 2 metric 10;
        }
        interface lo0.0;
    }
}
policy-options {
    policy-statement next-hop-self {
        term change-next-hop {
            from neighbor 10.1.12.1;
            then {
                next-hop self;
            }
        }
    }
    policy-statement import-toR1 {
        term 1 {
            from {
                route-filter 100.100.1.0/24 exact;
            }
            then {
                local-preference 200;
            }
        }
    }

user@R4> show configuration
[...Output truncated...]
interfaces {
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.1.46.1/30;
            }
            family iso;
        }
```

```
    }
    so-0/0/2 {
        unit 0 {
            family inet {
                address 10.1.45.1/30;
            }
            family iso;
        }
    }
    so-0/0/3 {
        unit 0 {
            family inet {
                address 10.1.24.2/30;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.0.4/32;
            }
            family iso {
                address 49.0001.1000.0000.0004.00;
            }
        }
    }
}
routing-options {
    [...Output truncated...]
    router-id 10.0.0.4;
    autonomous-system 65002;
}
protocols {
    bgp {
        group internal {
            type internal;
            local-address 10.0.0.4;
            export next-hop-self;
            neighbor 10.0.0.2;
            neighbor 10.0.0.3;
            neighbor 10.0.0.6;
        }
```

```
        group toR5 {
            type external;
            peer-as 65001;
            neighbor 10.1.45.2;
        }
    }
    isis {
        level 1 disable;
        interface all {
            level 2 metric 10;
        }
        interface lo0.0;
    }
}
policy-options {
    policy-statement next-hop-self {
        term change-next-hop {
            from neighbor 10.1.45.2;
            then {
                next-hop self;
            }
        }
    }
```

Meaning

The sample output shows a basic BGP configuration on border routers `R2` and `R4`. Both routers have the AS (65002) included at the [`routing-options`] hierarchy level. Each router has two groups included at the [`protocols bgp group` *group*] hierarchy level. External peers are included in the external group, either *toR1* or `toR5`, depending on the router. Internal peers are included in the `internal` group. The underlying IGP protocol is IS-IS on both routers, and relevant interfaces are configured to run IS-IS.

Note that in the configuration on both routers, the router ID is manually configured to avoid duplicate router ID problems, and the `next-hop-self` statement is included to avoid any BGP next-hop reachability problems.

Verify Advertised BGP Routes

IN THIS SECTION

- Purpose | 1490
- Action | 1490
- Meaning | 1490

Purpose

You can determine if a particular route that you have configured is being advertised to a neighbor.

Action

To verify the routing information as it has been prepared for advertisement to the specified BGP neighbor, enter the following Junos OS CLI operational mode command:

```
user@host> show route advertising-protocol bgp neighbor-address
```

Sample Output

command-name

```
user@R2> show route advertising-protocol bgp 10.0.0.4\
inet.0: 20 destinations, 22 routes (20 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 100.100.1.0/24          Self                 5       200        65001 I
* 100.100.2.0/24          Self                 5       100        65001 I
* 100.100.3.0/24          Self                         100        65001 I
* 100.100.4.0/24          Self                         100        65001 I
```

Meaning

The sample output shows the BGP routes advertised from `R2` to its neighbor, `10.0.0.4` (`R4`). Out of 22 total routes in the `inet.0` routing table, 20 are active destinations . No routes are `hidden` or in the `hold-down` state. Routes reside in the `hold-down` state prior to being declared active, and routes rejected by a routing

policy can be placed into the `hidden` state. The information displayed reflects the routes that the routing table exported to the BGP routing protocol.

Verify That a Particular BGP Route Is Received on Your Router

IN THIS SECTION

- Purpose | 1491
- Action | 1491
- Meaning | 1492

Purpose

Display the routing information as it is received through a particular BGP neighbor and advertised by the local router to the neighbor.

Action

To verify that a particular BGP route is received on your router, enter the following Junos OS CLI operational mode command:

```
user@host>                          show route receive-protocol bgp neighbor-
address
```

Sample Output

command-name

```
user@R6> show route receive-protocol bgp 10.0.0.2
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 100.100.1.0/24          10.0.0.2             5       200        65001 I
* 100.100.2.0/24          10.0.0.2             5       100        65001 I
  100.100.3.0/24          10.0.0.2                     100        65001 I
  100.100.4.0/24          10.0.0.2                     100        65001 I
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

```
user@R6> show route receive-protocol bgp 10.0.0.4
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 100.100.3.0/24           10.0.0.4                     100        65001 I
* 100.100.4.0/24          10.0.0.4                     100        65001 I
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Meaning

The sample output shows four BGP routes from R2 and two from R4. Of the four routes from R2, only two are active in the routing table, as indicated by the asterisk (*), while both routes received from R4 are active in the routing table. All BGP routes came through AS 65001.

Examine BGP Routes and Route Selection

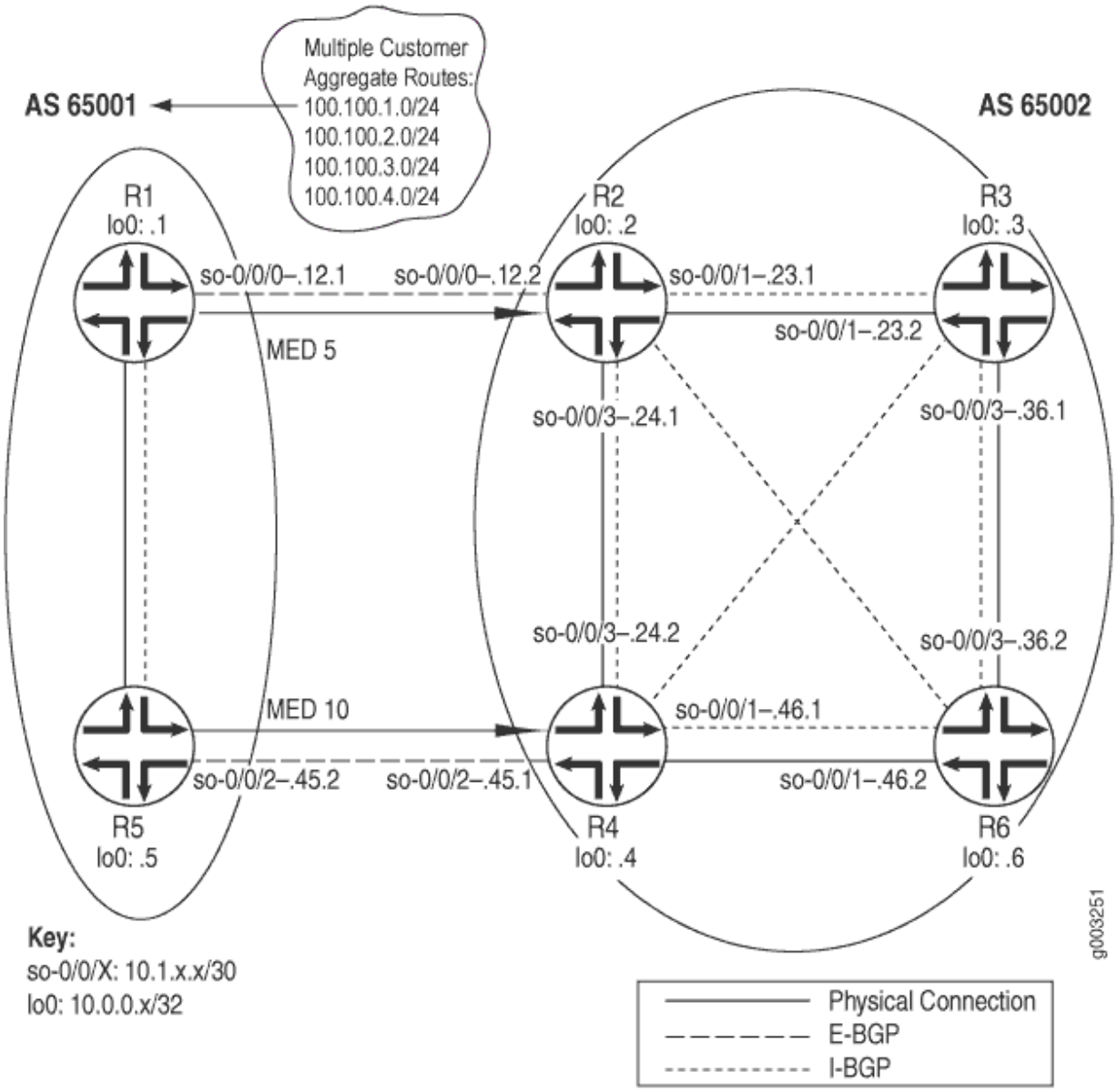
IN THIS SECTION

- Examine the Local Preference Selection | **1495**
- Examine the Multiple Exit Discriminator Route Selection | **1496**
- Examine the EBGP over IBGP Selection | **1498**
- Examine the IGP Cost Selection | **1500**

Purpose

You can examine the BGP path selection process to determine the single, active path when BGP receives multiple routes to the same destination prefix.

Figure 103: BGP Network Topology



The network in Figure 103 on page 1493 shows that R1 and R5 announce the same aggregate routes to R2 and R4, which results in R2 and R4 receiving two routes to the same destination prefix. The route selection process on R2 and R4 determines which of the two BGP routes received is active and advertised to the other internal routers (R3 and R6).

Before the router installs a BGP route, it must make sure that the BGP `next-hop` attribute is reachable. If the BGP next hop cannot be resolved, the route is not installed. When a BGP route is installed in the routing table, it must go through a path selection process if multiple routes exist to the same destination prefix. The BGP path selection process proceeds in the following order:

1. Route preference in the routing table is compared. For example, if an OSPF and a BGP route exist for a particular destination, the OSPF route is selected as the active route because the OSPF route has a default preference of 110, while the BGP route has a default preference of 170.
2. Routes are compared for local preference. The route with the highest local preference is preferred. For example, see "Examine the Local Preference Selection" on page 1476.
3. The AS path attribute is evaluated. The shorter AS path is preferred.
4. The origin code is evaluated. The lowest origin code is preferred (`I (IGP) < E (EGP) < ? (Incomplete)`).
5. The MED value is evaluated. By default, if any of the routes are advertised from the same neighboring AS, the lowest MED value is preferred. The absence of a MED value is interpreted as a MED of 0. For an example, see "Examine the Multiple Exit Discriminator Route Selection" on page 356.
6. The route is evaluated as to whether it is learned through EBGP or IBGP. EBGP learned routes are preferred to IBGP learned routes. For an example, see "Examine the EBGP over IBGP Selection" on page 1476
7. If the route is learned from IBGP, the route with the lowest IGP cost is preferred. For an example, see "Examine the IGP Cost Selection" on page 1476. The physical next hop to the IBGP peer is installed according to the following three rules:

    a. i. After BGP examines the `inet.0` and `inet.3` routing tables, the physical next hop of the route with the lowest preference is used.

    b. i. If the preference values in the `inet.0` and the `inet.3` routing tables are a tie, the physical next hop of the route in the `inet.3` routing table is used.

    c. i. When a preference tie exists in the same routing table, the physical next hop of the route with more paths is installed.
8. The route reflection cluster list attribute is evaluated. The shortest length cluster list is preferred. Routes without a cluster list are considered to have a cluster list length of 0.
9. The router ID is evaluated. The route from the peer with the lowest router ID is preferred (usually the loopback address).
10. The peer address value is examined. The peer with the lowest peer IP address is preferred.

To determine the single, active path when BGP receives multiple routes to the same destination prefix, enter the following Junos OS CLI operational mode command:

```
user@host> show route destination-prefix < detail >
```

The following steps illustrate the inactive reason displayed when BGP receives multiple routes to the same destination prefix and one route is selected as the single, active path:

Examine the Local Preference Selection

IN THIS SECTION

- Purpose | **1495**
- Action | **1495**
- Meaning | **1496**

Purpose

To examine a route to determine if local preference is the selection criteria for the single, active path.

Action

To examine a route to determine if local preference is the selection criteria for the single, active path, enter the following Junos OS CLI operational mode command:

```
user@host> show route destination-prefix < detail >
```

Sample Output

command-name

```
user@R4> show route 100.100.1.0 detail
inet.0: 20 destinations, 24 routes (20 active, 0 holddown, 0 hidden)
100.100.1.0/24 (2 entries, 1 announced)
         *BGP    Preference: 170/-201
                 Source: 10.0.0.2
                Next hop: 10.1.24.1 via so-0/0/3.0, selected
                Protocol next hop: 10.0.0.2 Indirect next hop: 8644000 277
                State: <Active Int Ext>
                Local AS: 65002 Peer AS: 65002
                Age: 2:22:34    Metric: 5       Metric2: 10
                Task: BGP_65002.10.0.0.2+179
```

```
                Announcement bits (3): 0-KRT 3-BGP.0.0.0.0+179 4-Resolve inet.0
                 AS path: 65001 I
                 Localpref: 200
                Router ID: 10.0.0.2
          BGP   Preference: 170/-101
                 Source: 10.1.45.2
                Next hop: 10.1.45.2 via so-0/0/2.0, selected
                State: <Ext>
                 Inactive reason: Local Preference
                Local AS: 65002 Peer AS: 65001
                Age: 2w0d 1:28:31       Metric: 10
                Task: BGP_65001.10.1.45.2+179
                 AS path: 65001 I
                Localpref: 100
                Router ID: 10.0.0.5
```

Meaning

The sample output shows that R4 received two instances of the 100.100.1.0 route: one from 10.0.0.2 (R2) and one from 10.1.45.2 (R5). R4 selected the path from R2 as its active path, as indicated by the asterisk (*). The selection is based on the local preference value contained in the Localpref field. The path with the *highest* local preference is preferred. In the example, the path with the higher local preference value is the path from R2, 200.

The reason that the route from R5 is not selected is in the Inactive reason field, in this case, Local Preference.

Note that the two paths are from the same neighboring network: AS 65001.

Examine the Multiple Exit Discriminator Route Selection

IN THIS SECTION

- Purpose | 1496
- Action | 1497
- Meaning | 1498

Purpose

To examine a route to determine if the MED is the selection criteria for the single, active path.

Action

To examine a route to determine if the MED is the selection criteria for the single, active path, enter the following Junos OS CLI operational mode command:

```
user@host> show route destination-prefix < detail >
```

Sample Output

command-name

```
user@R4> show route 100.100.2.0 detail
inet.0: 20 destinations, 24 routes (20 active, 0 holddown, 0 hidden)
100.100.2.0/24 (2 entries, 1 announced)
         *BGP     Preference: 170/-101
                  Source: 10.0.0.2
                 Next hop: 10.1.24.1 via so-0/0/3.0, selected
                 Protocol next hop: 10.0.0.2 Indirect next hop: 8644000 277
                 State: <Active Int Ext>
                 Local AS: 65002 Peer AS: 65002
                 Age: 2:32:01     Metric: 5       Metric2: 10
                 Task: BGP_65002.10.0.0.2+179
                 Announcement bits (3): 0-KRT 3-BGP.0.0.0.0+179 4-Resolve inet.0
                  AS path: 65001 I
                 Localpref: 100
                 Router ID: 10.0.0.2
         BGP     Preference: 170/-101
                  Source: 10.1.45.2
                 Next hop: 10.1.45.2 via so-0/0/2.0, selected
                 State: <NotBest Ext>
                  Inactive reason: Not Best in its group
                 Local AS: 65002 Peer AS: 65001
                 Age: 2w0d 1:37:58        Metric: 10
                 Task: BGP_65001.10.1.45.2+179
                  AS path: 65001 I
                 Localpref: 100
                 Router ID: 10.0.0.5
```

Meaning

The sample output shows that R4 received two instances of the 100.100.2.0 route: one from 10.0.0.2 (R2), and one from 10.1.45.2 (R5). R4 selected the path from R2 as its active route, as indicated by the asterisk (*). The selection is based on the MED value contained in the Metric: field. The path with the lowest MED value is preferred. In the example, the path with the lowest MED value (5) is the path from R2. Note that the two paths are from the same neighboring network: AS 65001.

The reason that the inactive path is not selected is displayed in the Inactive reason: field, in this case, Not Best in its group. The wording is used because the Junos OS uses the process of deterministic MED selection, by default.

Examine the EBGP over IBGP Selection

IN THIS SECTION

- Purpose | 1498
- Action | 1498
- Meaning | 1499

Purpose

To examine a route to determine if EBGP is selected over IBGP as the selection criteria for the single, active path.

Action

To examine a route to determine if EBGP is selected over IBGP as the selection criteria for the single, active path, enter the following Junos OS CLI operational mode command:

```
user@host> show route destination-prefix < detail >
```

Sample Output

command-name

```
user@R4> show route 100.100.3.0 detail
inet.0: 20 destinations, 24 routes (20 active, 0 holddown, 0 hidden)
100.100.3.0/24 (2 entries, 1 announced)
         *BGP     Preference: 170/-101
                  Source: 10.1.45.2
                 Next hop: 10.1.45.2 via so-0/0/2.0, selected
                 State: <Active Ext>
                  Local AS: 65002 Peer AS: 65001
                 Age: 5d 0:31:25
                 Task: BGP_65001.10.1.45.2+179
                 Announcement bits (3): 0-KRT 3-BGP.0.0.0.0+179 4-Resolve inet.0
                 AS path: 65001 I
                 Localpref: 100
                 Router ID: 10.0.0.5
         BGP     Preference: 170/-101
                  Source: 10.0.0.2
                 Next hop: 10.1.24.1 via so-0/0/3.0, selected
                 Protocol next hop: 10.0.0.2 Indirect next hop: 8644000 277
                 State: <NotBest Int Ext>
                  Inactive reason: Interior > Exterior > Exterior via Interior
                 Local AS: 65002 Peer AS: 65002
                 Age: 2:48:18    Metric2: 10
                 Task: BGP_65002.10.0.0.2+179
                 AS path: 65001 I
                 Localpref: 100
                 Router ID: 10.0.0.2
```

Meaning

The sample output shows that R4 received two instances of the 100.100.3.0 route: one from 10.1.45.2 (R5) and one from 10.0.0.2 (R2). R4 selected the path from R5 as its active path, as indicated by the asterisk (*). The selection is based on a preference for routes learned from an EBGP peer over routes learned from an IBGP. R5 is an EBGP peer.

You can determine if a path is received from an EBGP or IBGP peer by examining the Local As and Peer As fields. For example, the route from R5 shows the local AS is 65002 and the peer AS is 65001, indicating

that the route is received from an EBGP peer. The route from R2 shows that both the local and peer AS is 65002, indicating that it is received from an IBGP peer.

The reason that the inactive path is not selected is displayed in the Inactive reason field, in this case, Interior > Exterior > Exterior via Interior. The wording of this reason shows the order of preferences applied when the same route is received from two routers. The route received from a strictly internal source (IGP) is preferred first, the route received from an external source (EBGP) is preferred next, and any route which comes from an external source and is received internally (IBGP) is preferred last. Therefore, EBGP routes are selected over IBGP routes as the active path.

Examine the IGP Cost Selection

IN THIS SECTION

- Purpose | 1500
- Action | 1500
- Meaning | 1501

Purpose

To examine a route to determine if EBGP is selected over IBGP as the selection criteria for the single, active path.

Action

To examine a route to determine if EBGP is selected over IBGP as the selection criteria for the single, active path, enter the following Junos OS CLI operational mode command:

```
user@host> show route destination-prefix < detail >
```

Sample Output

command-name

```
user@R6> show route 100.100.4.0 detail
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
100.100.4.0/24 (2 entries, 1 announced)
```

```
        *BGP    Preference: 170/-101
                Source: 10.0.0.4
                Next hop: 10.1.46.1 via so-0/0/1.0, selected
                Protocol next hop: 10.0.0.4 Indirect next hop: 864c000 276
                State: <Active Int Ext>
                Local AS: 65002 Peer AS: 65002
                Age: 2:16:11    Metric2: 10
                Task: BGP_65002.10.0.0.4+4120
                Announcement bits (2): 0-KRT 4-Resolve inet.0
                AS path: 65001 I
                Localpref: 100
                Router ID: 10.0.0.4
         BGP    Preference: 170/-101
                Source: 10.0.0.2
                Next hop: 10.1.46.1 via so-0/0/1.0, selected
                Next hop: 10.1.36.1 via so-0/0/3.0
                Protocol next hop: 10.0.0.2 Indirect next hop: 864c0b0 278
                State: <NotBest Int Ext>
                Inactive reason: IGP metric
                Local AS: 65002 Peer AS: 65002
                Age: 2:16:03    Metric2: 20
                Task: BGP_65002.10.0.0.2+179
                AS path: 65001 I
                Localpref: 100
                Router ID: 10.0.0.2
```

Meaning

The sample output shows that `R6` received two instances of the `100.100.4.0` route: one from `10.0.0.4` (`R4`) and one from `10.0.0.2` (`R2`). `R6` selected the path from `R4` as its active route, as indicated by the asterisk (*). The selection is based on the IGP metric, displayed in the `Metric2` field. The route with the lowest IGP metric is preferred. In the example, the path with the lowest IGP metric value is the path from `R4`, with an IGP metric value of 10, while the path from `R2` has an IGP metric of 20. Note that the two paths are from the same neighboring network: AS 65001.

The reason that the inactive path was not selected is displayed in the `Inactive reason` field, in this case, `IGP metric`.

Checklist for Checking the BGP Layer

IN THIS SECTION

- Problem | 1502
- Solution | 1502

Problem

Description

This checklist provides the steps and commands for checking the BGP configuration of the Multiprotocol Label Switching (MPLS) network. The checklist provides links to an overview of the BGP configuration and more detailed information about the commands used to configure BGP. (See Table 18 on page 1502.)

Solution

Table 18: Checklist for Checking the BGP Layer

Tasks	Command or Action
"Checking the BGP Layer" on page 1503	
1. "Check That BGP Traffic Is Using the LSP" on page 1476	`traceroute` *`hostname`*
1. "Check BGP Sessions" on page 1476	`show bgp summary`
1. "Verify the BGP Configuration" on page 1476	`show configuration`
1. "Examine BGP Routes" on page 1476	`show route` *`destination-prefix`* `detail`
1. "Verify Received BGP Routes" on page 1476	`show route receive protocol bgp` *`neighbor-address`*

Table 18: Checklist for Checking the BGP Layer *(Continued)*

Tasks	Command or Action

1. "Taking Appropriate Action for Resolving the Network Problem" on page 1456	The following sequence of commands addresses the specific problem described in this topic:
	`[edit] edit protocols bgp`
	`[edit protocols bgp] show set local-address 10.0.0.1 delete group internal neighbor 10.1.36.2 show commit`

1. "Check That BGP Traffic Is Using the LSP Again" on page 1476	`traceroute hostname`

Checking the BGP Layer

IN THIS SECTION

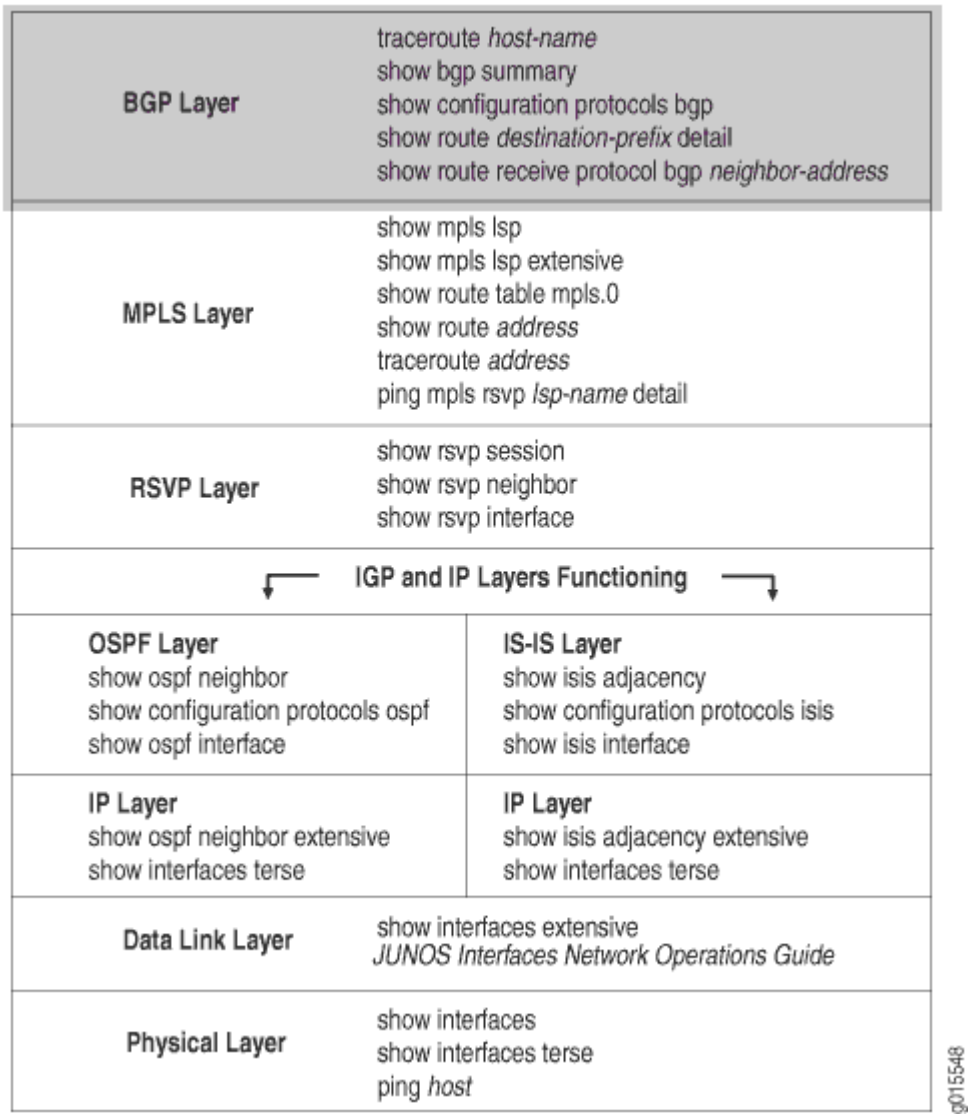
- Check That BGP Traffic Is Using the LSP | **1505**
- Check BGP Sessions | **1506**
- Verify the BGP Configuration | **1509**
- Examine BGP Routes | **1516**
- Verify Received BGP Routes | **1518**
- Taking Appropriate Action for Resolving the Network Problem | **1520**
- Check That BGP Traffic Is Using the LSP Again | **1521**

Purpose

After you have configured the label-switched path (LSP) and determined that it is up, and configured BGP and determined that sessions are established, ensure that BGP is using the LSP to forward traffic.

Figure 104 on page 1504 illustrates the BGP layer of the layered MPLS model.

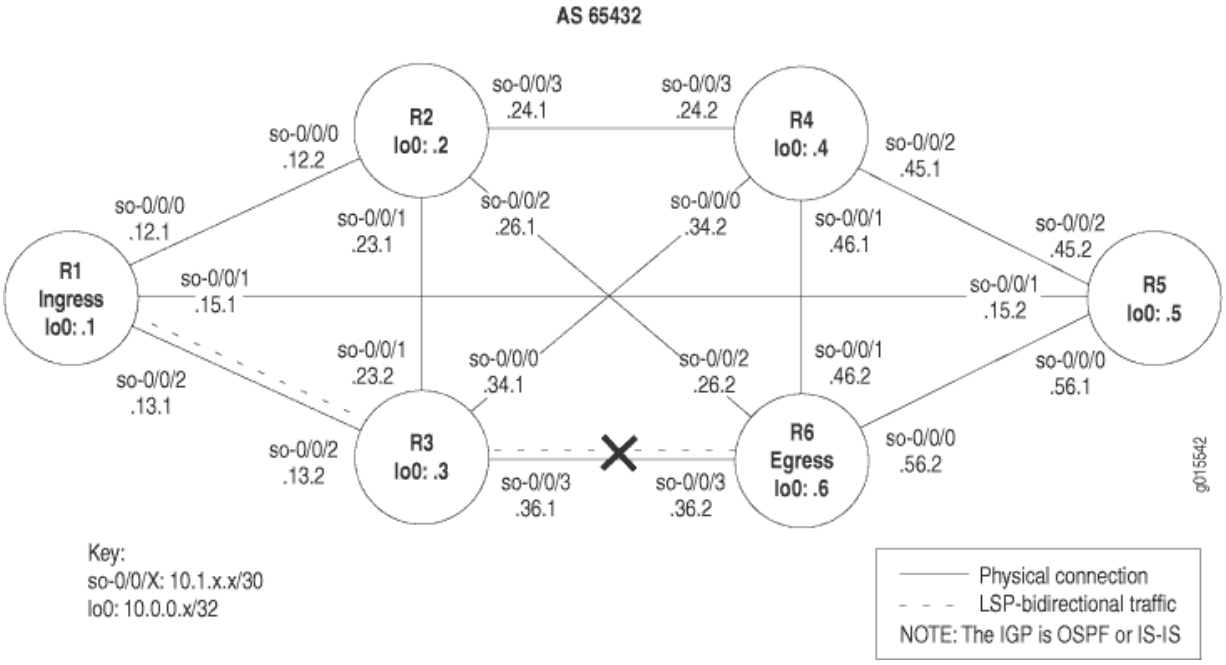
Figure 104: Checking the BGP Layer



When you check the BGP layer, you verify that the route is present and active, and more importantly, you ensure that the next hop is the LSP. There is no point in checking the BGP layer unless the LSP is established, because BGP uses the MPLS LSP to forward traffic. If the network is not functioning at the BGP layer, the LSP does not work as configured.

Figure 105 on page 1505 illustrates the MPLS network used in this topic.

Figure 105: MPLS Network Broken at the BGP Layer



The network shown in Figure 105 on page 1505 is a fully meshed configuration where every directly connected interface can receive and send packets to every other similar interface. The LSP in this network is configured to run from ingress router **R1**, through transit router **R3**, to egress router **R6**. In addition, a reverse LSP is configured to run from **R6** through **R3** to **R1**, creating bidirectional traffic.

The cross shown in Figure 105 on page 1505 indicates where BGP is not being used to forward traffic through the LSP. Possible reasons for the LSP not working correctly are that the destination IP address of the LSP does not equal the BGP next hop or that BGP is not configured properly.

To check the BGP layer, follow these steps:

Check That BGP Traffic Is Using the LSP

IN THIS SECTION

- Purpose | **1506**
- Action | **1506**
- Meaning | **1506**

Purpose

At this level of the troubleshooting model, BGP and the LSP may be up, however BGP traffic might not be using the LSP to forward traffic.

Action

To verify that BGP traffic is using the LSP, enter the following Junos OS command-line interface (CLI) operational mode command from the ingress router:

```
user@host> traceroute hostname
```

Sample Output

command-name

```
user@R1> traceroute 100.100.6.1
traceroute to 100.100.6.1 (100.100.6.1), 30 hops max, 40 byte packets
 1  10.1.13.2 (10.1.13.2)  0.653 ms  0.590 ms  0.543 ms
 2  10.1.36.2 (10.1.36.2)  0.553 ms !N  0.552 ms !N  0.537 ms !N

user@R6> traceroute 100.100.1.1
traceroute to 100.100.1.1 (100.100.1.1), 30 hops max, 40 byte packets
 1  10.1.36.1 (10.1.36.1)  0.660 ms  0.551 ms  0.526 ms
 2  10.1.13.1 (10.1.13.1)  0.568 ms !N  0.553 ms !N  0.536 ms !N
```

Meaning

The sample output shows that BGP traffic is not using the LSP, consequently MPLS labels do not appear in the output. Instead of using the LSP, BGP traffic is using the interior gateway protocol (IGP) to reach the BGP next-hop LSP egress address for **R6** and **R1**. The Junos OS default is to use LSPs for BGP traffic when the BGP next hop equals the LSP egress address.

Check BGP Sessions

IN THIS SECTION

Purpose | 1507

Action | 1507
Meaning | 1508

Purpose

Display summary information about BGP and its neighbors to determine if routes are received from peers in the autonomous system (AS). When a BGP session is established, the peers are exchanging update messages.

Action

To check that BGP sessions are up, enter the following Junos OS CLI operational mode command from the ingress router:

```
user@host>              show bgp summary
```

Sample Output 1

command-name

```
user@R1> show bgp summary
Groups: 1 Peers: 6  Down peers: 1
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 1          1          0          0          0          0
Peer              AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/Received/
Damped...
10.0.0.2        65432      11257      11259       0       0 3d 21:49:57  0/0/0
0/0/0
10.0.0.3        65432      11257      11259       0       0 3d 21:49:57 0/0/0
0/0/0
10.0.0.4        65432      11257      11259       0       0 3d 21:49:57 0/0/0
0/0/0
10.0.0.5        65432      11257      11260       0       0 3d 21:49:57 0/0/0
0/0/0
10.0.0.6        65432          4       4572       0       1 3d 21:46:59 Active
```

```
10.1.36.2       65432      11252      11257       0       0 3d 21:46:49  1/1/0
0/0/0
```

Sample Output 2

command-name

```
user@R1> show bgp summary
Groups: 1 Peers: 5 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0                 1          1          0          0          0          0
Peer              AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/Received/
Damped...
10.0.0.2        65432         64         68       0       0       32:18 0/0/0
0/0/0
10.0.0.3        65432         64         67       0       0       32:02 0/0/0
0/0/0
10.0.0.4        65432         64         67       0       0       32:10 0/0/0
0/0/0
10.0.0.5        65432         64         67       0       0       32:14 0/0/0
0/0/0
10.0.0.6        65432         38         39       0       1       18:02 1/1/0
0/0/0
```

Meaning

Sample Output 1 shows that one peer (egress router **10.0.0.6**) is not established, as indicated by the **Down Peers: 1** field. The last column (**State|#Active/Received/Damped**) shows that peer **10.0.0.6** is active, indicating that is it not established. All other peers are established as indicated by the number of active, received, and damped routes. For example, **0/0/0** for peer **10.0.0.2** indicates that no BGP routes were active or received in the routing table, and no BGP routes were damped; **1/1/0** for peer **10.1.36.2** indicates that one BGP route was active and received in the routing table, and no BGP routes were damped.

If the output of the `show bgp summary` command of an ingress router shows that a neighbor is down, check the BGP configuration. For information on checking the BGP configuration, see "Verify the BGP Configuration" on page 1476.

Sample Output 2 shows output from ingress router **R1** after the BGP configurations on **R1** and **R6** were corrected in "Taking Appropriate Action for Resolving the Network Problem" on page 1456.. All BGP peers are established and one route is active and received. No BGP routes were damped.

If the output of the `show bgp summary` command shows that a neighbor is up but packets are not being forwarded, check for received routes from the egress router. For information on checking the egress router for received routes, see "Verify Received BGP Routes" on page 1476.

Verify the BGP Configuration

IN THIS SECTION

- Purpose | **1509**
- Action | **1509**
- Meaning | **1516**

Purpose

For BGP to run on the router, you must define the local AS number, configure at least one group, and include information about at least one peer in the group (the peer's IP address and AS number). When BGP is part of an MPLS network, you must ensure that the LSP is configured with a destination IP address equal to the BGP next hop in order for BGP routes to be installed with the LSP as the next hop for those routes.

Action

To verify the BGP configuration, enter the following Junos OS CLI operational mode command:

```
user@host>                          show configuration
```

Sample Output 1

command-name

```
user@R1> show configuration
[...Output truncated...]
interfaces {
    so-0/0/0 {
        unit 0 {
            family inet {
                address 10.1.12.1/30;
```

```
            }
            family iso;
            family mpls;
        }
    }
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.1.15.1/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/0/2 {
        unit 0 {
            family inet {
                address 10.1.13.1/30;
            }
             family iso;
            family mpls;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 192.168.70.143/21;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.0.1/32;
            }
             family iso {
                address 49.0004.1000.0000.0001.00;
            }
        }
    }
}
routing-options {
    [...Output truncated...]
```

```
          route 100.100.1.0/24 reject;
    }
     router-id 10.0.0.1;
     autonomous-system 65432;
}
protocols {
    rsvp {
        interface so-0/0/0.0;
        interface so-0/0/1.0;
        interface so-0/0/2.0;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path R1-to-R6 {
            to  10.0.0.6; <<< destination address of the LSP
        }
        inactive: interface so-0/0/0.0;
        inactive: interface so-0/0/1.0;
        interface so-0/0/2.0;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        export send-statics; <<< missing local-address statement
         group internal {
            type internal;
            neighbor 10.0.0.2;
            neighbor 10.0.0.5;
            neighbor 10.0.0.4;
            neighbor 10.0.0.6;
            neighbor 10.0.0.3;
            neighbor 10.1.36.2; <<< incorrect interface address
        }
    }
    isis {
        level 1 disable;
        interface so-0/0/0.0;
        interface so-0/0/1.0;
        interface so-0/0/2.0;
        interface all {
```

```
            level 2 metric 10;
        }
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-0/0/0.0;
            interface so-0/0/1.0;
            interface so-0/0/2.0;
            interface lo0.0; {
                passive
            }
        }
    }
}
policy-options {
    policy-statement send-statics {
        term statics {
            from {
                route-filter 100.100.1.0/24 exact;
            }
            then accept;
        }
    }
}
```

Sample Output 2

command-name

```
user@R6> show configuration
[...Output truncated...]
interfaces {
    so-0/0/0 {
        unit 0 {
```

```
            family inet {
                address 10.1.56.2/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.1.46.2/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/0/2 {
        unit 0 {
            family inet {
                address 10.1.26.2/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/0/3 {
        unit 0 {
            family inet {
                address 10.1.36.2/30;
            }
             family iso;
            family mpls;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 192.168.70.148/21;
            }
        }
    }
    lo0 {
        unit 0 {
```

```
            family inet {
                address 10.0.0.6/32;
                address 127.0.0.1/32;
            }
            family iso {
                address 49.0004.1000.0000.0006.00;
            }
        }
    }
}
routing-options {
    [...Output truncated...]
         route 100.100.6.0/24 reject;
    }
     router-id 10.0.0.6;
     autonomous-system 65432;
}
protocols {
    rsvp {
        interface so-0/0/0.0;
        interface so-0/0/1.0;
        interface so-0/0/2.0;
        interface so-0/0/3.0;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path R6-to-R1 {
            to 10.0.0.1; <<< destination address of the reverse LSP
        }
        inactive: interface so-0/0/0.0;
        inactive: interface so-0/0/1.0;
        inactive: interface so-0/0/2.0;
        interface so-0/0/3.0;
    }
    bgp {
        group internal {
            type internal;
            export send-statics; <<< missing local-address statement
             neighbor 10.0.0.2;
            neighbor 10.0.0.3;
            neighbor 10.0.0.4;
```

```
            neighbor 10.0.0.5;
            neighbor 10.0.0.1;
            neighbor 10.1.13.1;         <<< incorrect interface address
        }
    }
    isis {
        level 1 disable;
        interface all {
            level 2 metric 10;
        }
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-0/0/0.0;
            interface so-0/0/1.0;
            interface so-0/0/2.0;
            interface so-0/0/3.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}
policy-options {
    policy-statement send-statics {
        term statics {
            from {
                route-filter 100.100.6.0/24 exact;
            }
            then accept;
        }
    }
}
```

Meaning

The sample output shows the BGP configurations on ingress router R1 and egress router R6. Both configurations show the local AS (65432), one group (internal), and six peers configured. The underlying interior gateway protocol is IS-IS, and the relevant interfaces are configured to run IS-IS.

NOTE: In this configuration, the RID is manually configured to avoid any duplicate RID problems, and all interfaces configured with BGP include the `family inet` statement at the [`edit interfaces` *type-fpc/pic/port* `unit` *logical-unit-number*] hierarchy level.

Sample output for ingress router R1 and egress router R6 shows that the BGP protocol configuration is missing the `local-address` statement for the internal group. When the `local-address` statement is configured, BGP packets are forwarded from the local router loopback (lo0) interface address, which is the address to which BGP peers are peering. If the `local-address` statement is not configured, BGP packets are forwarded from the outgoing interface address, which does not match the address to which BGP peers are peering, and BGP does not come up.

On the ingress router, the IP address (10.0.0.1) in the `local-address` statement should be the same as the address configured for the LSP on the egress router (R6) in the `to` statement at the [`edit protocols mpls label-switched-path` *lsp-path-name*] hierarchy level. BGP uses this address, which is identical to the LSP address, to forward BGP traffic through the LSP.

In addition, the BGP configuration on R1 includes two IP addresses for R6, an interface address (10.1.36.2) and a loopback (lo0) interface address (10.0.0.6), resulting in the LSP destination address (10.0.0.6) not matching the BGP next-hop address (10.1.36.2). The BGP configuration on R6 also includes two IP addresses for R1, an interface address (10.1.13.1) and a loopback (lo0) interface address, resulting in the reverse LSP destination address (10.0.0.1) not matching the BGP next-hop address (10.1.13.1).

In this instance, because the `local-address` statement is missing in the BGP configurations of both routers and the LSP destination address does not match the BGP next-hop address, BGP is not using the LSP to forward traffic.

Examine BGP Routes

IN THIS SECTION

- Purpose | **1517**
- Action | **1517**
- Meaning | **1518**

Purpose

You can examine the BGP path selection process to determine the single, active path when BGP receives multiple routes to the same destination. In this step, we examine the reverse LSP **R6-to-R1**, making **R6** the ingress router for that LSP.

Action

To examine BGP routes and route selection, enter the following Junos OS CLI operational mode command:

```
user@host>              show route destination-prefix detail
```

Sample Output 1

command-name

```
user@R6> show route 100.100.1.1 detail
inet.0: 30 destinations, 46 routes (29 active, 0 holddown, 1 hidden)
100.100.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Source: 10.1.13.1
                Next hop: via so-0/0/3.0, selected
                 Protocol next hop: 10.1.13.1  Indirect next hop: 8671594 304
                State: <Active Int Ext>
                Local AS: 65432 Peer AS: 65432
                Age: 4d 5:15:39         Metric2: 2
                Task: BGP_65432.10.1.13.1+3048
                Announcement bits (2): 0-KRT 4-Resolve inet.0
                AS path: I
                Localpref: 100
                 Router ID: 10.0.0.1
```

Sample Output 2

command-name

```
user@R6> show route 100.100.1.1 detail
inet.0: 30 destinations, 46 routes (29 active, 0 holddown, 1 hidden)
100.100.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Source: 10.0.0.1
                Next hop: via so-0/0/3.0 weight 1, selected
                 Label-switched-path R6-to-R1
                Label operation: Push 100000
                 Protocol next hop: 10.0.0.1  Indirect next hop: 8671330 301
                State: <Active Int Ext>
                Local AS: 65432 Peer AS: 65432
                Age: 24:35      Metric2: 2
                Task: BGP_65432.10.0.0.1+179
                Announcement bits (2): 0-KRT 4-Resolve inet.0
                AS path: I
                Localpref: 100
                 Router ID: 10.0.0.1
```

Meaning

Sample Output 1 shows that the BGP next hop (**10.1.13.1**) does not equal the LSP destination address (**10.0.0.1**) in the `to` statement at the [**edit protocols mpls label-switched-path** ***label-switched-path-name***] hierarchy level when the BGP configuration of **R6** and **R1** is incorrect.

Sample Output 2, taken after the configurations on R1 and R6 are corrected, shows that the BGP next hop (**10.0.0.1**) and the LSP destination address (**10.0.0.1**) are the same, indicating that BGP can use the LSP to forward BGP traffic.

Verify Received BGP Routes

IN THIS SECTION

- Purpose | **1519**
- Action | **1519**
- Meaning | **1520**

Purpose

Display the routing information received on router **R6**, the ingress router for the reverse LSP **R6-to-R1**.

Action

To verify that a particular BGP route is received on the egress router, enter the following Junos OS CLI operational mode command:

```
user@host>              show route receive protocol bgp neighbor-address
```

Sample Output 1

command-name

```
user@R6> show route receive-protocol bgp 10.0.0.1
inet.0: 30 destinations, 46 routes (29 active, 0 holddown, 1 hidden)
<<< missing route
inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

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

__juniper_private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Sample Output 2

command-name

```
user@R6> show route receive-protocol bgp 10.0.0.1
inet.0: 30 destinations, 46 routes (29 active, 0 holddown, 1 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 100.100.1.0/24          10.0.0.1                     100        I

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

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

__juniper_private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Meaning

Sample Output 1 shows that ingress router **R6** (reverse LSP **R6-to-R1**) does not receive any BGP routes into the **inet.0** routing table when the BGP configurations of **R1** and **R6** are incorrect.

Sample Output 2 shows a BGP route installed in the **inet.0** routing table after the BGP configurations on **R1** and **R6** are corrected using "Taking Appropriate Action for Resolving the Network Problem" on page 1456.

Taking Appropriate Action for Resolving the Network Problem

IN THIS SECTION

- Problem | **1520**
- Solution | **1520**

Problem

Description

The appropriate action depends on the type of problem you have isolated. In this example, a static route configured on `R2` is deleted from the [`routing-options`] hierarchy level. Other appropriate actions might include the following:

Solution

- Check the local router's configuration and edit it if appropriate.
- Troubleshoot the intermediate router.
- Check the remote host configuration and edit it if appropriate.
- Troubleshoot routing protocols.
- Identify additional possible causes.

To resolve the problem in this example, enter the following Junos OS CLI commands:

```
[edit]
                          user@R2# delete routing-options static route destination-
prefix
user@R2# commit and-quit
user@R2# show route destination-prefix
```

Sample Output

```
[edit]
user@R2# delete routing-options static route 10.0.0.5/32

[edit]
user@R2# commit and-quit
commit complete
Exiting configuration mode

user@R2> show route 10.0.0.5

inet.0: 22 destinations, 24 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[BGP/170] 3d 20:26:17, MED 5, localpref 100
                      AS path: 65001 I
                    > to 10.1.12.1 via so-0/0/0.0
```

Meaning

The sample output shows the static route deleted from the [routing-options] hierarchy and the new configuration committed. The output for the show route command now shows the BGP route as the preferred route, as indicated by the asterisk (*).

Check That BGP Traffic Is Using the LSP Again

IN THIS SECTION

- Purpose | 1522
- Action | 1522

- Meaning | 1522

Purpose

After taking the appropriate action to correct the error, the LSP needs to be checked again to confirm that BGP traffic is using the LSP and that the problem in the BGP layer has been resolved.

Action

To verify that BGP traffic is using the LSP, enter the following Junos OS CLI operational mode command from the ingress router:

```
user@host>              traceroute hostname
```

Sample Output

command-name

```
user@R1> traceroute 100.100.6.1
traceroute to 100.100.6.1 (100.100.6.1), 30 hops max, 40 byte packets
 1  10.1.13.2 (10.1.13.2)  0.858 ms  0.740 ms  0.714 ms
      MPLS Label=100016 CoS=0 TTL=1 S=1
 2  10.1.36.2 (10.1.36.2)  0.592 ms !N  0.564 ms !N  0.548 ms !N

user@R6> traceroute 100.100.1.1
traceroute to 100.100.1.1 (100.100.1.1), 30 hops max, 40 byte packets
 1  10.1.36.1 (10.1.36.1)  0.817 ms  0.697 ms  0.771 ms
      MPLS Label=100000 CoS=0 TTL=1 S=1
 2  10.1.13.1 (10.1.13.1)  0.581 ms !N  0.567 ms !N  0.544 ms !N
```

Meaning

The sample output shows that MPLS labels are used to forward packets through the LSP. Included in the output is a label value (**MPLS Label=100016**), the time-to-live value (**TTL=1**), and the stack bit value (**S=1**).

The **MPLS Label** field is used to identify the packet to a particular LSP. It is a 20-bit field, with a maximum value of (2^^20-1), approximately 1,000,000.

The time-to-live (TTL) value contains a limit on the number of hops that this MPLS packet can travel through the network (1). It is decremented at each hop, and if the TTL value drops below one, the packet is discarded.

The bottom of the stack bit value (**S=1**) indicates that is the last label in the stack and that this MPLS packet has one label associated with it. The MPLS implementation in the Junos OS supports a stacking depth of 3 on the M-series routers and up to 5 on the T-series routing platforms. For more information on MPLS label stacking, see RFC 3032, *MPLS Label Stack Encoding.*

MPLS labels appear in the sample output because the `traceroute` command is issued to a BGP destination where the BGP next hop for that route is the LSP egress address. The Junos OS by default uses LSPs for BGP traffic when the BGP next hop equals the LSP egress address.

If the BGP next hop does not equal the LSP egress address, the BGP traffic does not use the LSP, and consequently MPLS labels do not appear in the output for the `traceroute` command, as indicated in the sample output in "Check BGP Sessions" on page 1476.

Display Sent or Received BGP Packets

IN THIS SECTION

- Action | **1523**

Action

To configure the tracing for sent or received BGP protocol packets, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host# edit protocol bgp traceoptions
```

2. Configure the flag to display sent, received, or both sent and received packet information:

```
[edit protocols bgp traceoptions]
                user@host#  set flag update send
```

or

```
[edit protocols bgp traceoptions]
user@host# set flag update receive
```

or

```
[edit protocols bgp traceoptions]
user@host# set flag update
```

3. Verify the configuration:

```
user@host# show
```

For example:

```
[edit protocols bgp traceoptions]
user@host# show
file bgplog size 10k files 10;
flag update send;
```

or

```
[edit protocols bgp traceoptions]
user@host# show
file bgplog size 10k files 10;
flag update receive;
```

or

```
[edit protocols bgp traceoptions]
user@host# show
```

```
file bgplog size 10k files 10;
flag update send receive;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit protocols bgp traceoptions]
user@host# run show log bgplog
Sep 13 12:58:23 trace_on: Tracing to "/var/log/bgplog" started
Sep 13 12:58:23 BGP RECV flags 0x40 code ASPath(2): <null>
Sep 13 12:58:23 BGP RECV flags 0x40 code LocalPref(5): 100
Sep 13 12:58:23 BGP RECV flags 0xc0 code Extended Communities(16): 2:10458:3
[...Output truncated...]
```

Understanding Hidden Routes

Hidden routes are routes that the device cannot use for reasons such as an invalid next hop or a routing policy that rejects the routes.

NOTE: If a route is completely invalid, the route is not placed into the routing table as a candidate route and does not even appear as hidden.

Following are some useful commands for viewing and troubleshooting hidden routes:

- `show route hidden (terse | detail | extensive)`
- `show route hidden-route extensive`
- `show route next-hop-of-hidden-route extensive`
- `show route resolution unresolved detail`

Routes can be hidden for various reasons. A few of them are given below.

- An import policy rejects the route.
- The next hop cannot be resolved using the current indirect next hop resolution rule. Because routing protocols such as internal BGP (IBGP) can send routing information about indirectly connected routes, Junos OS relies on routes from intra-AS routing protocols (OSPF, IS-IS, RIP, and static) to resolve the best directly connected next hop. The Routing Engine performs route resolution to determine the best directly connected next hop and installs the route to the Packet Forwarding Engine.
- A damping policy suppresses the route.
- The AS path contains illegal or invalid confederation attributes.
- The next hop address is the address of the local routing device.
- The AS path contains illegal or invalid transitive attributes.
- The AS path is empty. This only applies to EBGP. For IBGP, an empty AS path is normal.
- The AS path contains a zero.
- The next hop address is a multicast address.
- The next hop address is an IPv6 link-local address.
- The route prefix or the route next hop is a martian address.
- The LDP (Label Distribution Protocol) session fails. The received routes are not installed in the routing table until the peer router reestablishes the LDP session.

SEE ALSO

Example: Configure IPv4 Static Routing for a Stub Network

Example: Configure IPv6 Static Routing for a Stub Network

Example: Optimizing Route Reconvergence by Enabling Indirect Next Hops on the Packet Forwarding Engine

Example: Configuring BGP Route Reflectors

Example: Configuring BGP Confederations | **1223**

Examples: Configuring BGP Flap Damping

Understand Basic Static Routing

Understanding BGP Confederations | **1221**

Understanding Indirect Next Hops

Examine Routes in the Forwarding Table

IN THIS SECTION

- Purpose | 1527
- Action | 1527
- Meaning | 1528

Purpose

When you run into problems, such as connectivity problems, you may need to examine routes in the forwarding table to verify that the routing protocol process has relayed the correct information into the forwarding table.

Action

To display the set of routes installed in the forwarding table, enter the following Junos OS CLI operational mode command:

```
user@host> show route forwarding-table
```

Sample Output

command-name

```
user@R2> show route forwarding-table
Routing table: inet
Internet:
Destination        Type RtRef Next hop           Type  Index NhRef Netif
default            perm     0                    rjct    10     1
10.0.0.2/32        intf     0 10.0.0.2           locl   256     1
10.0.0.3/32        user     1 10.1.23.0          ucst   282     4 so-0/0/1.0
10.0.0.4/32        user     1 10.1.24.0          ucst   290     7 so-0/0/3.0
10.0.0.6/32        user     1 10.1.24.0          ucst   290     7 so-0/0/3.0
10.1.12.0/30       intf     1 ff.3.0.21          ucst   278     6 so-0/0/0.0
```

```
10.1.12.0/32       dest     0 10.1.12.0          recv   280     1 so-0/0/0.0
10.1.12.2/32       intf     0 10.1.12.2          locl   277     1
10.1.12.3/32       dest     0 10.1.12.3          bcst   279     1 so-0/0/0.0
10.1.23.0/30       intf     0 ff.3.0.21          ucst   282     4 so-0/0/1.0
10.1.23.0/32       dest     0 10.1.23.0          recv   284     1 so-0/0/1.0
10.1.23.1/32       intf     0 10.1.23.1          locl   281     1
10.1.23.3/32       dest     0 10.1.23.3          bcst   283     1 so-0/0/1.0
10.1.24.0/30       intf     0 ff.3.0.21          ucst   290     7 so-0/0/3.0
10.1.24.0/32       dest     0 10.1.24.0          recv   292     1 so-0/0/3.0
10.1.24.1/32       intf     0 10.1.24.1          locl   289     1
10.1.24.3/32       dest     0 10.1.24.3          bcst   291     1 so-0/0/3.0
10.1.36.0/30       user     0 10.1.23.0          ucst   282     4 so-0/0/1.0
10.1.46.0/30       user     0 10.1.24.0          ucst   290     7 so-0/0/3.0
100.100.1.0/24     user     0 10.1.12.0          ucst   278     6 so-0/0/0.0
100.100.2.0/24     user     0 10.1.12.0          ucst   278     6 so-0/0/0.0
100.100.3.0/24     user     0 10.1.12.0          ucst   278     6 so-0/0/0.0
100.100.4.0/24     user     0 10.1.12.0          ucst   278     6 so-0/0/0.0
[...Output truncated...]
```

Meaning

The sample output shows the network-layer prefixes and their next hops installed in the forwarding table. The output includes the same next-hop information as in the `show route detail` command (the next-hop address and interface name). Additional information includes the destination type, the next-hop type, the number of references to this next hop, and an index into an internal next-hop database. (The internal database contains additional information used by the Packet Forwarding Engine to ensure proper encapsulation of packets sent out an interface. This database is not accessible to the user.

For detailed information about the meanings of the various flags and types fields, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers

IN THIS SECTION

Requirements | 1529

Overview | 1529
Configuration | 1530
Verification | 1532

This example shows how to override the default routing policy on packet transport routers, such as the PTX Series Packet Transport Routers.

Requirements

This example requires Junos OS Release 12.1 or later.

Overview

By default, the PTX Series routers do not install BGP routes in the forwarding table.

For PTX Series routers, the configuration of the `from protocols bgp` condition with the `then accept` action does not have the usual result that it has on other Junos OS routing devices. With the following routing policy on PTX Series routers, BGP routes do not get installed in the forwarding table.

```
user@host# show policy-options
policy-statement accept-no-install {
    term 1 {
        from protocol bgp;
        then accept;
    }
}
user@host# show routing-options
forwarding-table {
    export accept-no-install;
}
```

```
user@host> show route forwarding-table
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     2
```

No BGP routes are installed in the forwarding table. This is the expected behavior.

This example shows how to use the `then install-to-fib` action to effectively override the default BGP routing policy.

Configuration

IN THIS SECTION

- CLI Quick Configuration | 1530
- Installing Selected BGP Routes in the Forwarding Table | 1530

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.

```
set policy-options prefix-list install-bgp 66.0.0.1/32
set policy-options policy-statement override-ptx-series-default term 1 from prefix-list install-bgp
set policy-options policy-statement override-ptx-series-default term 1 then load-balance per-prefix
set policy-options policy-statement override-ptx-series-default term 1 then install-to-fib
set routing-options forwarding-table export override-ptx-series-default
```

Installing Selected BGP Routes in the Forwarding Table

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 Junos OS CLI User Guide.

To install selected BGP routes in the forwarding table:

1. Configure a list of prefixes to install in the forwarding table.

```
[edit policy-options prefix-list install-bgp]
user@host# set 66.0.0.1/32
```

2. Configure the routing policy, applying the prefix list as a condition.

```
[edit policy-options policy-statement override-ptx-series-default term 1]
user@host# set from prefix-list install-bgp
user@host# set then install-to-fib
user@host# set then load-balance per-prefix
```

3. Apply the routing policy to the forwarding table.

```
[edit routing-options forwarding-table]
user@host# set export override-ptx-series-default
```

Results

From configuration mode, confirm your configuration by entering the `show policy-options` 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@host# show policy-options
prefix-list install-bgp {
    66.0.0.1/32;
}
policy-statement override-ptx-series-default {
    term 1 {
        from {
            prefix-list install-bgp;
        }
        then {
            load-balance per-prefix;
            install-to-fib;
        }
```

```
    }
}
```

```
user@host# show routing-options
forwarding-table {
    export override-ptx-series-default;
}
```

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

Verification

IN THIS SECTION

- Verifying That the Selected Route Is Installed in the Forwarding Table | **1532**

Confirm that the configuration is working properly.

Verifying That the Selected Route Is Installed in the Forwarding Table

Purpose

Make sure that the configured policy overrides the default policy.

Action

From operational mode, enter the `show route forwarding-table` command.

```
user@host> show route forwarding-table destination 66.0.0.1
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
66.0.0.1/32        user     0                    indr 2097159     3
                                                 ulst 2097156     2
                            5.1.0.2              ucst   574     1 et-6/0/0.1
                            5.2.0.2              ucst   575     1 et-6/0/0.2
```

Meaning

This output shows that the route to 66.0.0.1/32 is installed in the forwarding table.

SEE ALSO

Basic BGP Routing Policies | 430

Log BGP State Transition Events

IN THIS SECTION

- Purpose | 1533
- Action | 1533
- Meaning | 1534

Purpose

Border Gateway Protocol (BGP) state transitions indicate a network problem and need to be logged and investigated.

Action

To log BGP state transition events to the system log, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
                user@host# edit protocol bgp
```

2. Configure the system log:

```
user@host#  set log-updown
```

3. Verify the configuration:

```
user@host# show
```

4. Commit the configuration:

```
user@host# commit
```

Meaning

Log messages from BGP state transition events are sufficient to diagnose most BGP session problems. Table 19 on page 1534 lists and describes the six states of a BGP session.

Table 19: Six States of a BGP Session

BGP State	Description
Idle	This is the first state of a connection. BGP waits for a start event initiated by an administrator. The start event might be the establishment of a BGP session through router configuration or the resetting of an existing session. After the start event, BGP initializes its resources, resets a connect-retry timer, initiates a TCP transport connection, and starts listening for connections initiated by remote peers. BGP then transitions to a **Connect** state. If there are errors, BGP falls back to the **Idle** state.
Connect	BGP waits for the transport protocol connection to complete. If the TCP transport connection is successful, the state transitions to **OpenSent**. If the transport connection is not successful, the state transitions to **Active**. If the connect-retry timer has expired, the state remains in the **Connect** state, the timer is reset, and a transport connection is initiated. With any other event, the state goes back to **Idle**.

Table 19: Six States of a BGP Session *(Continued)*

BGP State	Description
Active	BGP tries to acquire a peer by initiating a transport protocol connection. If it is successful, the state transitions to **OpenSent**. If the connect-retry timer expires, BGP restarts the connect timer and falls back to the **Connect** state. BGP continues to listen for a connection that may be initiated from another peer. The state may go back to **Idle** in case of other events, such as a stop event. In general, a neighbor state flip-flopping between **Connect** and **Active** is an indication that there is a problem with the TCP transport connection. Such a problem might be caused by many TCP retransmissions or the inability of a neighbor to reach the IP address of its peer.
OpenSent	BGP receives an open message from its peer. In the **OpenSent** state, BGP compares its autonomous system (AS) number with the AS number of its peer and recognizes whether the peer belongs to the same AS (internal BGP) or to a different AS (external BGP). The open message is checked for correctness. In case of errors, such as a bad version number of an unacceptable AS, BGP sends an error-notification message and goes back to **Idle**. For any other errors, such as expiration of the hold timer or a stop event, BGP sends a notification message with the corresponding error code and falls back to the **Idle** state. If there are no errors, BGP sends keepalive messages and resets the keepalive timer. In this state, the hold time is negotiated. If the hold time is 0, the hold and keepalive timers are not restarted. When a TCP transport disconnect is detected, the state falls back to **Active**.
OpenConfirm	BGP waits for a keepalive or notification message. If a keepalive is received, the state becomes **Established**, and the neighbor negotiation is complete. If the system receives an update or keepalive message, it restarts the hold timer (assuming that the negotiated hold time is not 0). If a notification message is received, the state falls back to **Idle**. The system sends periodic keepalive messages at the rate set by the keepalive timer. In case of a transport disconnect notification or in response to a stop event, the state falls back to **Idle**. In response to other events, the system sends a notification message with a finite state machine (FSM) error code and goes back to **Idle**.

Table 19: Six States of a BGP Session *(Continued)*

BGP State	Description
Established	This is the final state in the neighbor negotiation. In this state, BGP exchanges update ackets with its peers and the hold timer is restarted at the receipt of an update or keepalive message when it is not set to zero. If the system receives a notification message, the state falls back to **Idle**. Update messages are checked for errors, such as missing attributes, duplicate attributes, and so on. If errors are found, a notification is sent to the peer, and the state falls back to **Idle**. BGP goes back to **Idle** when the hold timer expires, a disconnect notification is received from the transport protocol, a stop event is received, or in response to any other event.

For more detailed BGP protocol packet information, configure BGP-specific tracing. See "Checklist for Tracking Error Conditions" on page 1466 for more information.

Configure BGP-Specific Options

IN THIS SECTION

- Display Detailed BGP Protocol Information | **1537**
- Diagnose BGP Session Establishment Problems | **1539**

Purpose

When unexpected events or problems occur, or if you want to diagnose BGP establishment issues, you can view more detailed information by configuring options specific to BGP. You can also configure tracing for a specific BGP peer or peer group. For more information, see the *Junos System Basics Configuration Guide*.

Display Detailed BGP Protocol Information

IN THIS SECTION

- Action | 1537
- Meaning | 1538

Action

To display BGP protocol information in detail, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host#  edit protocol bgp traceoptions
```

2. Configure the flag to display detailed BGP protocol messages:

```
[edit protocols bgp traceoptions]
user@host#  set flag update detail
```

3. Verify the configuration:

```
user@host# show
```

For example:

```
[edit protocols bgp traceoptions]
user@host# show
flag update detail;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit protocols bgp traceoptions]
user@pro5-a# run show log bgp
Sep 17 14:47:16 trace_on: Tracing to "/var/log/bgp" started
Sep 17 14:47:17 bgp_read_v4_update: receiving packet(s) from 10.255.245.53 (Internal AS 10458)
Sep 17 14:47:17 BGP RECV 10.255.245.53+179 -> 10.255.245.50+1141
Sep 17 14:47:17 BGP RECV message type 2 (Update) length 128
Sep 17 14:47:17 BGP RECV flags 0x40 code Origin(1): IGP
Sep 17 14:47:17 BGP RECV flags 0x40 code ASPath(2): 2
Sep 17 14:47:17 BGP RECV flags 0x80 code MultiExitDisc(4): 0
Sep 17 14:47:17 BGP RECV flags 0x40 code LocalPref(5): 100
Sep 17 14:47:17 BGP RECV flags 0xc0 code Extended Communities(16): 2:10458:1
[...Output truncated...]
```

Meaning

Table 4 lists tracing flags specific to BGP and presents example output for some of the flags. You can also configure tracing for a specific BGP peer or peer group. For more information, see the *Junos System Basics Configuration Guide*.

Table 20: BGP Protocol Tracing Flags

Tracing Flags	Description	Example Output
aspath	AS path regular expression operations	Not available.
damping	Damping operations	**Nov 28 17:01:12 bgp_damp_change: Change event Nov 28 17:01:12 bgp_dampen: Damping 10.10.1.0 Nov 28 17:01:12 bgp_damp_change: Change event Nov 28 17:01:12 bgp_dampen: Damping 10.10.2.0 Nov 28 17:01:12 bgp_damp_change: Change event Nov 28 17:01:12 bgp_dampen: Damping 10.10.3.0**

Table 20: BGP Protocol Tracing Flags *(Continued)*

Tracing Flags	Description	Example Output
keepalive	BGP keepalive messages	**Nov 28 17:09:27 bgp_send: sending 19 bytes to 10.217.5.101 (External AS 65471) Nov 28 17:09:27 Nov 28 17:09:27 BGP SEND 10.217.5.1+179 -> 10.217.5.101+52162 Nov 28 17:09:27 BGP SEND message type 4 (KeepAlive) length 19 Nov 28 17:09:28 Nov 28 17:09:28 BGP RECV 10.217.5.101+52162 -> 10.217.5.1+179 Nov 28 17:09:28 BGP RECV message type 4 (KeepAlive) length 19**
open	BGP open packets	**Nov 28 18:37:42 bgp_send: sending 37 bytes to 10.217.5.101 (External AS 65471) Nov 28 18:37:42 Nov 28 18:37:42 BGP SEND 10.217.5.1+179 -> 10.217.5.101+38135 Nov 28 18:37:42 BGP SEND message type 1 (Open) length 37**
packets	All BGP protocol packets	**Sep 27 17:45:31 BGP RECV 10.0.100.108+179 -> 10.0.100.105+1033 Sep 27 17:45:31 BGP RECV message type 4 (KeepAlive) length 19 Sep 27 17:45:31 bgp_send: sending 19 bytes to 10.0.100.108 (Internal AS 100) Sep 27 17:45:31 BGP SEND 10.0.100.105+1033 -> 10.0.100.108+179 Sep 27 17:45:31 BGP SEND message type 4 (KeepAlive) length 19 Sep 27 17:45:31 bgp_read_v4_update: receiving packet(s) from 10.0.100.108 (Internal AS 100)**
update	Update packets	**Nov 28 19:05:24 BGP SEND 10.217.5.1+179 -> 10.217.5.101+55813 Nov 28 19:05:24 BGP SEND message type 2 (Update) length 53 Nov 28 19:05:24 bgp_send: sending 65 bytes to 10.217.5.101 (External AS 65471) Nov 28 19:05:24 Nov 28 19:05:24 BGP SEND 10.217.5.1+179 -> 10.217.5.101+55813 Nov 28 19:05:24 BGP SEND message type 2 (Update) length 65 Nov 28 19:05:24 bgp_send: sending 55 bytes to 10.217.5.101 (External AS 65471)**

Diagnose BGP Session Establishment Problems

IN THIS SECTION

- Purpose | 1540
- Action | 1540

Purpose

To trace BGP session establishment problems.

Action

To trace BGP session establishment problems, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
                user@host#  edit protocol bgp
```

2. Configure BGP open messages:

```
[edit protocols bgp]
user@host#  set traceoptions flag open detail
```

3. Verify the configuration:

```
user@host# show
```

For example:

```
[edit protocols bgp]
user@host# show
traceoptions {
        file bgplog size 10k files 10;
        flag open detail;
}
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host#run show log  filename
```

For example:

```
[edit protocols bgp]
user@hotst# run show log bgplog
```

```
Sep 17 17:13:14 trace_on: Tracing to "/var/log/bgplog" started
Sep 17 17:13:14 bgp_read_v4_update: done with 201.0.0.2 (Internal AS 10458) received 19
octets 0 updates 0 routes
Sep 17 17:13:15 bgp_read_v4_update: receiving packet(s) from 201.0.0.3 (Internal AS 10458)
Sep 17 17:13:15 bgp_read_v4_update: done with 201.0.0.3 (Internal AS 10458) received 19
octets 0 updates 0 routes
Sep 17 17:13:44 bgp_read_v4_update: receiving packet(s) from 201.0.0.2 (Internal AS 10458)
[...Output truncated...]
```

Configure IS-IS-Specific Options

IN THIS SECTION

- Displaying Detailed IS-IS Protocol Information | **1542**
- Displaying Sent or Received IS-IS Protocol Packets | **1545**
- Analyzing IS-IS Link-State PDUs in Detail | **1547**

Purpose

When unexpected events or problems occur, or if you want to diagnose IS-IS adjacency establishment issues, you can view more detailed information by configuring options specific to IS-IS.

To configure IS-IS options, follow these steps:

Displaying Detailed IS-IS Protocol Information

IN THIS SECTION

- Action | 1542
- Meaning | 1543

Action

To trace IS-IS messages in detail, follow these steps:

1. Configure the flag to display detailed IS-IS protocol messages.

```
[edit protocols isis traceoptions]
user@host#  set flag hello detail
```

2. Verify the configuration.

```
user@host# show
```

For example:

```
[edit protocols isis traceoptions]
user@host# show
file isislog size 10k files 10;
flag hello detail;
```

3. Commit the configuration.

```
user@host# commit
```

4. View the contents of the file containing the detailed messages.

```
user@host# run show log filename
```

For example:

```
user@host# run show log isislog
```

```
Nov 29 23:17:50 trace_on: Tracing to "/var/log/isislog" started
Nov 29 23:17:50 Sending PTP IIH on so-1/1/1.0
Nov 29 23:17:53 Sending PTP IIH on so-1/1/0.0
Nov 29 23:17:54 Received PTP IIH, source id abc-core-01 on so-1/1/0.0
Nov 29 23:17:54     from interface index 11
Nov 29 23:17:54     max area 0, circuit type l2, packet length 4469
Nov 29 23:17:54     hold time 30, circuit id 6
Nov 29 23:17:54     neighbor state up
Nov 29 23:17:54     speaks IP
Nov 29 23:17:54     area address 99.0008 (1)
Nov 29 23:17:54     IP address 10.10.10.29
Nov 29 23:17:54     4396 bytes of total padding
Nov 29 23:17:54     updating neighbor abc-core-01
Nov 29 23:17:55 Received PTP IIH, source id abc-core-02 on so-1/1/1.0
Nov 29 23:17:55     from interface index 12
Nov 29 23:17:55     max area 0, circuit type l2, packet length 4469
Nov 29 23:17:55     hold time 30, circuit id 6
Nov 29 23:17:55     neighbor state up
Nov 29 23:17:55     speaks IP
Nov 29 23:17:55     area address 99.0000 (1)
Nov 29 23:17:55     IP address 10.10.10.33
Nov 29 23:17:55     4396 bytes of total padding
Nov 29 23:17:55     updating neighbor abc-core-02
```

Meaning

Table 5 lists tracing flags that can be configured specific to IS-IS and presents example output for some of the flags.

Table 21: IS-IS Protocol Tracing Flags

Tracing Flags	Description	Example Output
csn	Complete sequence number PDU (CSNP)	**Nov 28 20:02:48 Sending L2 CSN on interface so-1/1/0.0Nov 28 20:02:48 Sending L2 CSN on interface so-1/1/1.0** With the **detail** option. **Nov 28 20:06:08 Sending L2 CSN on interface so-1/1/1.0Nov 28 20:06:08 LSP abc-core-01.00-00 lifetime 1146Nov 28 20:06:08 sequence 0x1c4f8 checksum 0xa1e9Nov 28 20:06:08 LSP abc-core-02.00-00 lifetime 411Nov 28 20:06:08 sequence 0x7435 checksum 0x5424Nov 28 20:06:08 LSP abc-brdr-01.00-00 lifetime 465Nov 28 20:06:08 sequence 0xf73 checksum 0xab10Nov 28 20:06:08 LSP abc-edge-01.00-00 lifetime 1089Nov 28 20:06:08 sequence 0x1616 checksum 0xdb29Nov 28 20:06:08 LSP abc-edge-02.00-00 lifetime 1103Nov 28 20:06:08 sequence 0x45cc checksum 0x6883**
hello	Hello packet	**Nov 28 20:13:50 Sending PTP IIH on so-1/1/1.0Nov 28 20:13:50 Received PTP IIH, source id abc-core-01 on so-1/1/0.0Nov 28 20:13:53 Received PTP IIH, source id abc-core-02 on so-1/1/1.0Nov 28 20:13:57 Sending PTP IIH on so-1/1/0.0Nov 28 20:13:58 Received PTP IIH, source id abc-core-01 on so-1/1/0.0Nov 28 20:13:59 Sending PTP IIH on so-1/1/1.0**
lsp	Link-state PDUs (LSPs)	**Nov 28 20:15:46 Received L2 LSP abc-edge-01.00-00, interface so-1/1/0.0Nov 28 20:15:46 from abc-core-01Nov 28 20:15:46 sequence 0x1617, checksum 0xd92a, lifetime 1197Nov 28 20:15:46 Updating L2 LSP abc-edge-01.00-00 in TEDNov 28 20:15:47 Received L2 LSP abc-edge-01.00-00, interface so-1/1/1.0Nov 28 20:15:47 from abc-core-02Nov 28 20:15:47 sequence 0x1617, checksum 0xd92a, lifetime 1197**
lsp-generation	Link-state PDU generation packets	**Nov 28 20:21:24 Regenerating L1 LSP abc-edge-03.00-00, old sequence 0x682Nov 28 20:21:27 Rebuilding L1, fragment abc-edge-03.00-00Nov 28 20:21:27 Rebuilt L1 fragment abc-edge-03.00-00, size 59Nov 28 20:31:52 Regenerating L2 LSP abc-edge-03.00-00, old sequence 0x689Nov 28 20:31:54 Rebuilding L2, fragment abc-edge-03.00-00Nov 28 20:31:54 Rebuilt L2 fragment abc-edge-03.00-00, size 256Nov 28 20:34:05 Regenerating L1 LSP abc-edge-03.00-00, old sequence 0x683Nov 28 20:34:08 Rebuilding L1, fragment abc-edge-03.00-00Nov 28 20:34:08 Rebuilt L1 fragment abc-edge-03.00-00, size 59**
packets	All IS-IS protocol packets	Not available.

Table 21: IS-IS Protocol Tracing Flags *(Continued)*

Tracing Flags	Description	Example Output
psn	Partial sequence number PDU (PSNP) packets	**Nov 28 20:40:39 Received L2 PSN, source abc-core-01, interface so-1/1/0.0Nov 28 20:40:39 Received L2 PSN, source abc-core-02, interface so-1/1/1.0Nov 28 20:41:36 Sending L2 PSN on interface so-1/1/1.0Nov 28 20:41:36 Sending L2 PSN on interface so-1/1/0.0Nov 28 20:42:35 Received L2 PSN, source abc-core-02, interface so-1/1/1.0Nov 28 20:42:35 LSP abc-edge-03.00-00 lifetime 1196Nov 28 20:42:35 sequence 0x68c checksum 0x746dNov 28 20:42:35 Received L2 PSN, source abc-core-01, interface so-1/1/0.0Nov 28 20:42:35 LSP abc-edge-03.00-00 lifetime 1196Nov 28 20:42:35 sequence 0x68c checksum 0x746dNov 28 20:42:49 Sending L2 PSN on interface so-1/1/1.0Nov 28 20:42:49 LSP abc-core-01.00-00 lifetime 1197Nov 28 20:42:49 sequence 0x1c4fb checksum 0x9becNov 28 20:42:49 Sending L2 PSN on interface so-1/1/0.0Nov 28 20:42:49 LSP abc-core-01.00-00 lifetime 1197Nov 28 20:42:49 sequence 0x1c4fb checksum 0x9bec**
spf	Shortest-path-first (SPF) calculations	**Nov 28 20:44:01 Scheduling SPF for L1: ReconfigNov 28 20:44:01 Scheduling multicast SPF for L1: ReconfigNov 28 20:44:01 Scheduling SPF for L2: ReconfigNov 28 20:44:01 Scheduling multicast SPF for L2: ReconfigNov 28 20:44:02 Running L1 SPFNov 28 20:44:02 L1 SPF initialization complete: 0.000099s cumulative timeNov 28 20:44:02 L1 SPF primary processing complete: 0.000303s cumulative timeNov 28 20:44:02 L1 SPF result postprocessing complete: 0.000497s cumulative timeNov 28 20:44:02 L1 SPF RIB postprocessing complete: 0.000626s cumulative timeNov 28 20:44:02 L1 SPF routing table postprocessing complete: 0.000736s cumulative time**

SEE ALSO

> *Understanding IS-IS Areas to Divide an Autonomous System into Smaller Groups*
>
> *Example: Configuring a Multi-Level IS-IS Topology to Control Interarea Flooding*

Displaying Sent or Received IS-IS Protocol Packets

To configure the tracing for only sent or received IS-IS protocol packets, follow these steps:

1. Configure the flag to display sent, received, or both sent and received packets.

```
[edit protocols isis traceoptions]
user@host# set flag hello send
```

or

```
[edit protocols isis traceoptions]
user@host# set flag hello receive
```

or

```
[edit protocols isis traceoptions]
user@host# set flag hello
```

2. Verify the configuration.

```
user@host# show
```

For example:

```
[edit protocols isis traceoptions]
user@host# show
file isislog size 10k files 10;
flag hello send;
```

or

```
[edit protocols isis traceoptions]
user@host# show
file isislog size 10k files 10;
flag hello receive;
```

or

```
[edit protocols isis traceoptions]
user@host# show
```

```
file isislog size 10k files 10;
flag hello send receive;
```

3. Commit the configuration.

```
user@host# commit
```

4. View the contents of the file containing the detailed messages.

```
user@host# run show log filename
```

For example:

```
user@host# run show log isislog
Sep 27 18:17:01 ISIS periodic xmit to 01:80:c2:00:00:15 (IFL 2)
Sep 27 18:17:01 ISIS periodic xmit to 01:80:c2:00:00:14 (IFL 2)
Sep 27 18:17:03 ISIS periodic xmit to 01:80:c2:00:00:15 (IFL 2)
Sep 27 18:17:04 ISIS periodic xmit to 01:80:c2:00:00:14 (IFL 2)
Sep 27 18:17:06 ISIS L2 hello from 0000.0000.0008 (IFL 2) absorbed
Sep 27 18:17:06 ISIS periodic xmit to 01:80:c2:00:00:15 (IFL 2)
Sep 27 18:17:06 ISIS L1 hello from 0000.0000.0008 (IFL 2) absorbed
```

SEE ALSO

> *Understanding IS-IS Areas to Divide an Autonomous System into Smaller Groups*
>
> *Example: Configuring a Multi-Level IS-IS Topology to Control Interarea Flooding*

Analyzing IS-IS Link-State PDUs in Detail

To analyze IS-IS link-state PDUs in detail, follow these steps:

1. Configure IS-IS open messages.

```
[edit protocols isis traceoptions]
user@host# set flag lsp detail
```

2. Verify the configuration.

```
user@host# show
```

For example:

```
[edit protocols isis traceoptions]
user@host# show
file isislog size 5m world-readable;
flag error;
flag lsp detail;
```

3. Commit the configuration.

```
user@host# commit
```

4. View the contents of the file containing the detailed messages.

```
user@host# run show log filename
```

For example:

```
user@host# run show log isislog
Nov 28 20:17:24 Received L2 LSP abc-core-01.00-00, interface so-1/1/0.0
Nov 28 20:17:24     from abc-core-01
Nov 28 20:17:24     sequence 0x1c4f9, checksum 0x9fea, lifetime 1199
Nov 28 20:17:24     max area 0, length 426
Nov 28 20:17:24     no partition repair, no database overload
Nov 28 20:17:24     IS type 3, metric type 0
Nov 28 20:17:24     area address 99.0908 (1)
Nov 28 20:17:24     speaks CLNP
Nov 28 20:17:24     speaks IP
Nov 28 20:17:24     dyn hostname abc-core-01
Nov 28 20:17:24     IP address 10.10.134.11
Nov 28 20:17:24    IP prefix: 10.10.10.0/30 metric 1 up
Nov 28 20:17:24    IP prefix: 10.10.10.4/30 metric 5 up
Nov 28 20:17:24    IP prefix: 10.10.10.56/30 metric 5 up
Nov 28 20:17:24    IP prefix: 10.10.10.52/30 metric 1 up
Nov 28 20:17:24    IP prefix: 10.10.10.64/30 metric 5 up
```

```
Nov 28 20:17:24    IP prefix: 10.10.10.20/30 metric 5 up
Nov 28 20:17:24    IP prefix: 10.10.10.28/30 metric 5 up
Nov 28 20:17:24    IP prefix: 10.10.10.44/30 metric 5 up
Nov 28 20:17:24     IP prefix 10.10.10.0 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 1
Nov 28 20:17:24     IP prefix 10.10.10.4 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.10.10.56 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.10.10.52 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 1
Nov 28 20:17:24     IP prefix 10.10.10.64 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.10.10.20 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.10.10.28 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.10.10.44 255.255.255.252
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IS neighbors:
Nov 28 20:17:24     IS neighbor abc-core-02.00
Nov 28 20:17:24     internal, metrics: default 1
[...Output truncated...]
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IS neighbor abc-brdr-01.00
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IS neighbor abc-core-02.00, metric: 1
Nov 28 20:17:24     IS neighbor abc-esr-02.00, metric: 5
Nov 28 20:17:24     IS neighbor abc-edge-03.00, metric: 5
Nov 28 20:17:24     IS neighbor abc-edge-01.00, metric: 5
Nov 28 20:17:24     IS neighbor abc-edge-02.00, metric: 5
Nov 28 20:17:24     IS neighbor abc-brdr-01.00, metric: 5
Nov 28 20:17:24    IP prefix: 10.10.134.11/32 metric 0 up
Nov 28 20:17:24    IP prefix: 10.11.0.0/16 metric 5 up
Nov 28 20:17:24    IP prefix: 10.211.0.0/16 metric 0 up
Nov 28 20:17:24     IP prefix 10.10.134.11 255.255.255.255
Nov 28 20:17:24     internal, metrics: default 0
Nov 28 20:17:24     IP prefix 10.11.0.0 255.255.0.0
Nov 28 20:17:24     internal, metrics: default 5
Nov 28 20:17:24     IP prefix 10.211.0.0 255.255.0.0
Nov 28 20:17:24     internal, metrics: default 0
Nov 28 20:17:24     Updating LSP
Nov 28 20:17:24 Updating L2 LSP abc-core-01.00-00 in TED
```

```
Nov 28 20:17:24 Analyzing subtlv's for abc-core-02.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24 Analyzing subtlv's for abc-esr-02.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24 Analyzing subtlv's for abc-edge-03.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24 Analyzing subtlv's for abc-edge-01.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24 Analyzing subtlv's for abc-edge-02.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24 Analyzing subtlv's for abc-brdr-01.00
Nov 28 20:17:24 Analysis complete
Nov 28 20:17:24     Scheduling L2 LSP abc-core-01.00-00 sequence 0x1c4f9 on interface
so-1/1/1.0
```

SEE ALSO

Understanding IS-IS Areas to Divide an Autonomous System into Smaller Groups

Example: Configuring a Multi-Level IS-IS Topology to Control Interarea Flooding

Configure OSPF-Specific Options

IN THIS SECTION

- Diagnose OSPF Session Establishment Problems | **1551**
- Analyze OSPF Link-State Advertisement Packets in Detail | **1556**

Purpose

When unexpected events or problems occur, or if you want to diagnose OSPF neighbor establishment issues, you can view more detailed information by configuring options specific to OSPF.

To configure OSPF options, follow these steps:

Diagnose OSPF Session Establishment Problems

IN THIS SECTION

- Action | 1551
- Meaning | 1552

Action

To trace OSPF messages in detail, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host# edit protocols ospf traceoptions
```

2. Configure OSPF hello messages:

```
[edit protocols ospf traceoptions]
user@host# set flag hello detail
```

3. Verify the configuration:

```
user@host# show
```

For example:

```
[edit protocols ospf traceoptions]
user@host# show
file ospf size 5m world-readable;
flag hello detail;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
user@host# run show log ospf
```

```
Dec  2 16:14:24   Version 2, length 44, ID 10.0.0.6, area 1.0.0.0
Dec  2 16:14:24   checksum 0xf01a, authtype 0
Dec  2 16:14:24   mask 0.0.0.0, hello_ivl 10, opts 0x2, prio 128
Dec  2 16:14:24   dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0
Dec  2 16:14:24 OSPF sent Hello (1) -> 224.0.0.5 (so-1/1/2.0)
Dec  2 16:14:24   Version 2, length 44, ID 10.0.0.6, area 1.0.0.0
Dec  2 16:14:24   checksum 0xf01a, authtype 0
Dec  2 16:14:24   mask 0.0.0.0, hello_ivl 10, opts 0x2, prio 128
Dec  2 16:14:24   dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0
Dec  2 16:14:26 OSPF rcvd Hello 10.10.10.33 -> 224.0.0.5 (so-1/1/1.0)
Dec  2 16:14:26   Version 2, length 48, ID 10.10.134.12, area 0.0.0.0
Dec  2 16:14:26   checksum 0x99b8, authtype 0Dec  2 16:14:26   mask 255.255.255.252,
hello_ivl 10, opts 0x2, prio 1
ec  2 16:14:26   dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0
Dec  2 16:14:29 OSPF rcvd Hello 10.10.10.29 -> 224.0.0.5 (so-1/1/0.0)
Dec  2 16:14:29   Version 2, length 48, ID 10.108.134.11, area 0.0.0.0
Dec  2 16:14:29   checksum 0x99b9, authtype 0Dec  2 16:14:29   mask 255.255.255.252,
hello_ivl 10, opts 0x2, prio 1
Dec  2 16:14:29   dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0
```

Meaning

Table 6 lists OSPF tracing flags and presents example output for some of the flags.

Table 22: OSPF Protocol Tracing Flags

Tracing Flags	Description	Example Output
database-descripttion	All database description packets	**Dec 2 15:44:51 RPD_OSPF_NBRDOWN: OSPF neighbor 10.10.10.29 (so-1/1/0.0) state changed from Full to Down Dec 2 15:44:51 RPD_OSPF_NBRDOWN: OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Full to Down Dec 2 15:44:55 RPD_OSPF_NBRUP: OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Init to ExStart Dec 2 15:44:55 OSPF sent DbD (2) -> 224.0.0.5 (so-1/1/1.0) Dec 2 15:44:55 Version 2, length 32, ID 10.0.0.6, area 0.0.0.0 Dec 2 15:44:55 checksum 0xf76b, authtype 0 Dec 2 15:44:55 options 0x42, i 1, m 1, ms 1, seq 0xa009eee, mtu 4470 Dec 2 15:44:55 OSPF rcvd DbD 10.10.10.33 -> 224.0.0.5 (so-1/1/1.0) Dec 2 15:44:55 Version 2, length 32, ID 10.10.134.12, area 0.0.0.0 Dec 2 15:44:55 checksum 0x312c, authtype 0 Dec 2 15:44:55 options 0x42, i 1, m 1, ms 1, seq 0x2154, mtu 4470**
error	OSPF errored packets	**Dec 2 15:49:34 OSPF packet ignored: no matching interface from 172.16.120.29 Dec 2 15:49:44 OSPF packet ignored: no matching interface from 172.16.120.29 Dec 2 15:49:54 OSPF packet ignored: no matching interface from 172.16.120.29 Dec 2 15:50:04 OSPF packet ignored: no matching interface from 172.16.120.29 Dec 2 15:50:14 OSPF packet ignored: no matching interface from 172.16.120.29**

Table 22: OSPF Protocol Tracing Flags *(Continued)*

Tracing Flags	Description	Example Output
event	OSPF state transitions	**Dec 2 15:52:35 OSPF interface ge-2/2/0.0 state changed from DR to DR Dec 2 15:52:35 OSPF interface ge-3/1/0.0 state changed from DR to DR Dec 2 15:52:35 OSPF interface ge-3/2/0.0 state changed from DR to DR Dec 2 15:52:35 OSPF interface ge-4/2/0.0 state changed from DR to DR Dec 2 15:53:21 OSPF neighbor 10.10.10.29 (so-1/1/0.0) state changed from Full to Down Dec 2 15:53:21 RPD_OSPF_NBRDOWN: OSPF neighbor 10.10.10.29 (so-1/1/0.0) state changed from Full to Down Dec 2 15:53:21 OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Full to Down Dec 2 15:53:21 RPD_OSPF_NBRDOWN: OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Full to Down Dec 2 15:53:25 OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Down to Init Dec 2 15:53:25 OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Init to ExStart Dec 2 15:53:25 RPD_OSPF_NBRUP: OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Init to ExStart Dec 2 15:53:25 OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from ExStart to Exchange Dec 2 15:53:25 OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Exchange to Full Dec 2 15:53:25 RPD_OSPF_NBRUP: OSPF neighbor 10.10.10.33 (so-1/1/1.0) state changed from Exchange to Full**
flooding	Link-state flooding packets	**Dec 2 15:55:21 OSPF LSA Summary 10.218.0.0 10.0.0.6 flooding on so-1/1/0.0 Dec 2 15:55:21 OSPF LSA Summary 10.218.0.0 10.0.0.6 flooding on so-1/1/1.0 Dec 2 15:55:21 OSPF LSA Summary 10.218.0.0 10.0.0.6 on no so-1/1/2.0 rexmit lists, no flood Dec 2 15:55:21 OSPF LSA Summary 10.218.0.0 10.0.0.6 on no so-1/1/3.0 rexmit lists, no flood** **Dec 2 15:55:21 OSPF LSA Summary 10.245.0.1 10.0.0.6 on no so-1/1/2.0 rexmit lists, no flood Dec 2 15:55:21 OSPF LSA Summary 10.245.0.1 10.0.0.6 on no so-1/1/3.0 rexmit lists, no flood**

Table 22: OSPF Protocol Tracing Flags *(Continued)*

Tracing Flags	Description	Example Output
hello	Hello packets	**Dec 2 15:57:25 OSPF sent Hello (1) -> 224.0.0.5 (ge-3/1/0.0) Dec 2 15:57:25 Version 2, length 44, ID 10.0.0.6, area 2.0.0.0 Dec 2 15:57:25 checksum 0xe43f, authtype 0 Dec 2 15:57:25 mask 255.255.0.0, hello_ivl 10, opts 0x2, prio 128 Dec 2 15:57:25 dead_ivl 40, DR 10.218.0.1, BDR 0.0.0.0 Dec 2 15:57:25 OSPF rcvd Hello 10.10.10.33 -> 224.0.0.5 (so-1/1/1.0) Dec 2 15:57:25 Version 2, length 48, ID 10.10.134.12, area 0.0.0.0 Dec 2 15:57:25 checksum 0x99b8, authtype 0 Dec 2 15:57:25 mask 255.255.255.252, hello_ivl 10, opts 0x2, prio 1 Dec 2 15:57:25 dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0 Dec 2 15:57:27 OSPF sent Hello (1) -> 224.0.0.5 (ge-3/2/0.0) Dec 2 15:57:27 Version 2, length 44, ID 10.0.0.6, area 2.0.0.0 Dec 2 15:57:27 checksum 0xe4a5, authtype 0 Dec 2 15:57:27 mask 255.255.0.0, hello_ivl 10, opts 0x2, prio 128 Dec 2 15:57:27 dead_ivl 40, DR 10.116.0.1, BDR 0.0.0.0 Dec 2 15:57:28 OSPF rcvd Hello** 10.10.**10.29 -> 224.0.0.5 (so-1/1/0.0) Dec 2 15:57:28 Version 2, length 48, ID** 10.10.**134.11, area 0.0.0.0 Dec 2 15:57:28 checksum 0x99b9, authtype 0 Dec 2 15:57:28 mask 255.255.255.252, hello_ivl 10, opts 0x2, prio 1 Dec 2 15:57:28 dead_ivl 40, DR 0.0.0.0, BDR 0.0.0.0**
lsa-ack	Link-state acknowledgment packets	**Dec 2 16:00:11 OSPF rcvd LSAck 10.10.10.29 -> 224.0.0.5 (so-1/1/0.0) Dec 2 16:00:11 Version 2, length 44, ID** 10.10.**134.11, area 0.0.0.0 Dec 2 16:00:11 checksum 0xcdbf, authtype 0 Dec 2 16:00:11 OSPF rcvd LSAck 10.10.10.33 -> 224.0.0.5 (so-1/1/1.0) Dec 2 16:00:11 Version 2, length 144, ID** 10.10.**134.12, area 0.0.0.0 Dec 2 16:00:11 checksum 0x73bc, authtype 0 Dec 2 16:00:16 OSPF rcvd LSAck 10.10.10.33 -> 224.0.0.5 (so-1/1/1.0) Dec 2 16:00:16 Version 2, length 44, ID** 10.10.**134.12, area 0.0.0.0 Dec 2 16:00:16 checksum 0x8180, authtype 0**
lsa-request	Link-state request packets	**Dec 2 16:01:38 OSPF rcvd LSReq 10.10.10.29 -> 224.0.0.5 (so-1/1/0.0) Dec 2 16:01:38 Version 2, length 108, ID** 10.10.**134.11, area 0.0.0.0 Dec 2 16:01:38 checksum 0xe86, authtype 0**

Table 22: OSPF Protocol Tracing Flags *(Continued)*

Tracing Flags	Description	Example Output
lsa-update	Link-state update packets	**Dec 2 16:09:12 OSPF built router LSA, area 0.0.0.0 Dec 2 16:09:12 OSPF built router LSA, area 1.0.0.0 Dec 2 16:09:12 OSPF built router LSA, area 2.0.0.0 Dec 2 16:09:13 OSPF sent LSUpdate (4) -> 224.0.0.5 (so-1/1/0.0) Dec 2 16:09:13 Version 2, length 268, ID 10.0.0.6, area 0.0.0.0 Dec 2 16:09:13 checksum 0x8047, authtype 0 Dec 2 16:09:13 adv count 7 Dec 2 16:09:13 OSPF sent LSUpdate (4) -> 224.0.0.5 (so-1/1/1.0) Dec 2 16:09:13 Version 2, length 268, ID 10.0.0.6, area 0.0.0.0 Dec 2 16:09:13 checksum 0x8047, authtype 0 Dec 2 16:09:13 adv count 7**
packets	All OSPF packets	Not available.
packet-dump	Dump the contents of selected packet types	Not available.
spf	SPF calculations	**Dec 2 16:08:03 OSPF full SPF refresh scheduled Dec 2 16:08:04 OSPF SPF start, area 1.0.0.0 Dec 2 16:08:04 OSPF add LSA Router 10.0.0.6 distance 0 to SPF list Dec 2 16:08:04 SPF elapsed time 0.000525s Dec 2 16:08:04 Stub elapsed time 0.000263s Dec 2 16:08:04 OSPF SPF start, area 2.0.0.0 Dec 2 16:08:04 OSPF add LSA Router 10.0.0.6 distance 0 to SPF list Dec 2 16:08:04 SPF elapsed time 0.000253s Dec 2 16:08:04 Stub elapsed time 0.000249s Dec 2 16:08:04 OSPF SPF start, area 0.0.0.0 Dec 2 16:08:04 OSPF add LSA Router 10.0.0.6 distance 0 to SPF list Dec 2 16:08:04 OSPF add LSA Router** 10.10.**134.11 distance 1 to SPF list Dec 2 16:08:04 IP nexthop so-1/1/0.0 0.0.0.0 Dec 2 16:08:04 OSPF add LSA Router** 10.10.**134.12 distance 1 to SPF list Dec 2 16:08:04 IP nexthop so-1/1/1.0 0.0.0.0**

Analyze OSPF Link-State Advertisement Packets in Detail

IN THIS SECTION

- Action | **1557**

Action

To analyze OSPF link-state advertisement packets in detail, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host# edit protocols ospf traceoptions
```

2. Configure OSPF link-state packages:

```
[edit protocols ospf traceoptions]
user@host# set flag lsa-update detail
```

3. Verify the configuration:

```
user@host# show
```

   For example:

```
[edit protocols ospf traceoptions]
user@host# show
file ospf size 5m world-readable;
flag hello detail;
flag lsa-update detail;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
user@host#                  run show log ospf
```

```
Dec  2 16:23:47 OSPF sent LSUpdate (4) -> 224.0.0.5 (so-1/1/0.0) ec  2 16:23:47   Version 2,
length 196, ID 10.0.0.6, area 0.0.0.0
Dec  2 16:23:47   checksum 0xcc46, authtype 0
Dec  2 16:23:47   adv count 6 Dec  2 16:23:47 OSPF sent LSUpdate (4) -> 224.0.0.5
(so-1/1/1.0)
Dec  2 16:23:47   Version 2, length 196, ID 10.0.0.6, area 0.0.0.0 Dec  2 16:23:47   checksum
0xcc46, authtype 0
Dec  2 16:23:47   adv count 6
```

14

CHAPTER

Configuration Statements and Operational Commands

IN THIS CHAPTER

- peer-auto-discovery | **1560**
- Junos CLI Reference Overview | **1561**

peer-auto-discovery

IN THIS SECTION

- Syntax | 1560
- Hierarchy Level | 1560
- Description | 1561
- Required Privilege Level | 1561
- Release Information | 1561

Syntax

```
peer-auto-discovery {
    family inet6 {
        ipv6-nd;
    }
    interface interface_name;
}
```

Hierarchy Level

```
[edit fabric protocols bgp group <name> dynamic-neighbor dyn-name],
[edit logical-systems <name> protocols bgp group <name> dynamic-neighbor dyn-name],
[edit logical-systems routing-instances <name> protocols bgp group <name> dynamic-neighbor dyn-name],
[edit logical-systems <name> tenants <name> routing-instances <name> protocols bgp group <name> dynamic-neighbor dyn-name],
[edit protocols bgp group <name> dynamic-neighbor dyn-name],
[edit tenants <name> routing-instances <name> protocols bgp group <name> dynamic-neighbor dyn-name]
```

Description

Configure auto-discovery options for BGP neighbors to enable peering for a given interface or set of interfaces without specifying the local or remote neighbor addresses.

Required Privilege Level

routing

Release Information

Statement introduced in Junos OS 21.1R1.

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Read this guide to learn about the syntax and options that make up the statements and commands. Also understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands