JUNIPER NETWORKS

Junos® OS

Routing Policy Feature Guide for Routing Devices

Release
13.2

Published: 2013-07-31

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

Junos® OS Routing Policy Feature Guide for Routing Devices
13.2

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About the Documentation xv
Documentation and Release Notes xv
Supported Platforms xv
Using the Examples in This Manual xvi
Merging a Full Example xvi
Merging a Snippet xvii
Documentation Conventions xvii
Documentation Feedback xix
Requesting Technical Support xix
Self-Help Online Tools and Resources xix
Opening a Case with JTAC xx

Part 1 Overview

Chapter 1 Introduction to Routing Policy 3

Policy Framework Overview 3
Routing Policy and Firewall Filters 3
Reasons to Create a Routing Policy 4
Router Flows Affected by Policies 4
Control Points 7
Policy Components 8
Comparison of Routing Policies and Firewall Filters 8
Understanding Routing Policies 12
Importing and Exporting Routes 12
Active and Inactive Routes 13
Explicitly Configured Routes 14
Dynamic Database 14
Default Routing Policies 15

Chapter 2 Routing Policy Evaluation 17

How a Routing Policy Is Evaluated 17
Understanding Policy Expressions 19
Policy Expression Examples 20
Policy Expression Evaluation 21
Example: Evaluating Policy Expressions 22

Chapter 3 Route Filters 25

Understanding Route Filters for Use in Routing Policy Match Conditions 25
Radix Trees 25
Configuring Route Filters 27

How Route Filters Are Evaluated in Routing Policy Match Conditions 33
How an Address Mask Match Type Is Evaluated 34
How Prefix Order Affects Route Filter Evaluation 35
Common Configuration Problem with the Longest-Match Lookup 35
Route Filter Examples 35
Example: Rejecting Routes with Specific Destination Prefixes and Mask Lengths 36
Example: Rejecting Routes with a Mask Length Greater than Eight 36
Example: Rejecting Routes with Mask Length Between 26 and 29 37
Example: Rejecting Routes from Specific Hosts 37
Example: Accepting Routes with a Defined Set of Prefixes 38
Example: Rejecting Routes with a Defined Set of Prefixes 38
Example: Rejecting Routes with Prefixes Longer than 24 Bits 38
Example: Rejecting PIM Multicast Traffic Joins 39
Example: Rejecting PIM Traffic 39
Example: Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix 40
Example: Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths 41
Example: Evaluation of an Address Mask Match Type with Longest-Match Lookup 42

Chapter 4 Prefix Lists 45

Understanding Prefix Lists for Use in Routing Policy Match Conditions 45
Configuring Prefix Lists 46
How Prefix Lists Are Evaluated in Routing Policy Match Conditions 47
Configuring Prefix List Filters 47

Chapter 5 Policy Chains 49

How a Routing Policy Chain Is Evaluated 49

Chapter 6 Subroutines 51

Understanding Policy Subroutines in Routing Policy Match Conditions 51
Configuring Subroutines 51
Possible Consequences of Termination Actions in Subroutines 52
How a Routing Policy Subroutine Is Evaluated 54

Chapter 7 AS Paths 57

Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions 57
Configuring AS Path Regular Expressions 57
Configuring a Null AS Path 61
How AS Path Regular Expressions Are Evaluated 62
Examples: Configuring AS Path Regular Expressions 62
Understanding Prepending AS Numbers to BGP AS Paths 63
Understanding Adding AS Numbers to BGP AS Paths 64

Chapter 8 Communities 67
Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions 67
Understanding How to Define BGP Communities and Extended Communities 68
Defining BGP Communities for Use in Routing Policy Match Conditions 68
Using UNIX Regular Expressions in Community Names 70
Defining BGP Extended Communities for Use in Routing Policy Match Conditions 72
Examples: Defining BGP Extended Communities 73
How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions 74
Multiple Matches 75
Inverting Community Matches 76
Extended Community Type 76
Multiple Communities Are Matched with Ex-OR Logic 77
Including BGP Communities and Extended Communities in Routing Policy Match Conditions 78
Chapter 9 Testing Policies 79
Understanding Routing Policy Tests 79
Example: Testing a Routing Policy 79
Chapter 10 Damp BGP Route Flapping 81
Understanding Damping Parameters 81
Using Routing Policies to Damp BGP Route Flapping 82
Configuring BGP Flap Damping Parameters 83
Specifying BGP Flap Damping as the Action in Routing Policy Terms 85
Disabling Damping for Specific Address Prefixes 85
Example: Disabling Damping for a Specific Address Prefix 86
Example: Configuring BGP Flap Damping 86
Chapter 11 Source Class Usage and Destination Class Usage 89
Understanding Source Class Usage and Destination Class Usage Options 89
Chapter 12 Conditional Routing Policies 91
Understanding Conditional Installation of Prefixes in a Routing Table 92
Conditional Installation of Prefixes Use Cases 94
Chapter 13 Dynamic Routing Policies 97
Understanding Dynamic Routing Policies 97
Configuring Routing Policies and Policy Objects in the Dynamic Database 97
Configuring Routing Policies Based on Dynamic Database Configuration . . . 98
Applying Dynamic Routing Policies to BGP 100
Preventing Reestablishment of BGP Peering Sessions After NSR Routing Engine Switchover 100
Chapter 14 Discard Routing Policy 103
Understanding Forwarding Packets to the Discard Interface 103

Chapter 15 Reference Tables 105
Categories of Routing Policy Match Conditions 105
Routing Policy Match Conditions 107
Route Filter Match Conditions 115
Summary of Routing Policy Actions 117
Actions in Routing Policy Terms 119
Configuring Flow Control Actions 120
Configuring Actions That Manipulate Route Characteristics 121
Configuring the Default Action in Routing Policies 127
Example: Configuring the Default Action in a Routing Policy 128
Configuring a Final Action in Routing Policies 128
Logging Matches to a Routing Policy Term 129
Configuring Separate Actions for Routes in Route Lists 129
Protocol Support for Import and Export Policies 130

Part 2 Configuration

Chapter 16 Routing Policy Evaluation 133
Example: Applying Routing Policies at Different Levels of the BGP Hierarchy . . . 133
Example: Configuring a Conditional Default Route Policy 142
Example: Using Routing Policy in an ISP Network 149
Example: Disabling Suppression of Route Advertisements 197
Example: Configuring BGP to Advertise the Best External Route to Internal Peers 204
Example: Setting BGP to Advertise Inactive Routes 212
Example: Rejecting Known Invalid Routes 218
Example: Using Routing Policy to Set a Preference Value for BGP Routes 223

Chapter 17 Route Filters 229
Example: Configuring Policy Chains and Route Filters 229
Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF 240
Example: Configuring the MED Using Route Filters 245
Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters 257

Chapter 18 Prefix Lists 261
Example: Configuring Routing Policy Prefix Lists 261

Chapter 19 Subroutines 273
Example: Configuring a Policy Subroutine 273

Chapter 20 AS Paths 283
Example: Using AS Path Regular Expressions 283
Example: Configuring a Routing Policy to Prepend the AS Path 292
Example: Advertising Multiple Paths in BGP 301

Chapter 21 Communities 327
Example: Configuring Communities in a Routing Policy 327
Example: Configuring Extended Communities in a Routing Policy 342

Example: Defining a Routing Policy Based on the Number of BGP Communities . . . 351
Example: Defining a Routing Policy That Removes BGP Communities . . . 358

Chapter 22 Testing Policies . . . 367

Example: Testing a Routing Policy with Complex Regular Expressions . . . 367

Chapter 23 Damp BGP Route Flapping . . . 375

Example: Configuring Damping Parameters . . . 375
Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family . . . 384

Chapter 24 Source Class Usage and Destination Class Usage . . . 395

Example: Grouping Source and Destination Prefixes into a Forwarding Class . . 395

Chapter 25 Conditional Routing Policies . . . 405

Example: Configuring Conditional Installation of Prefixes in a Routing Table . . . 405

Chapter 26 Dynamic Routing Policies . . . 421

Example: Configuring Dynamic Routing Policies . . . 421

Chapter 27 Discard Routing Policy . . . 435

Example: Forwarding Packets to the Discard Interface . . . 435

Chapter 28 Routing Policy Configuration Statements . . . 445

address-family . . . 445
aigp-originate . . . 446
apply-path . . . 447
as-path (Policy Options) . . . 448
as-path-group . . . 449
ccc (Routing Policy Condition) . . . 450
community (Policy Options) . . . 451
condition . . . 454
damping (Policy Options) . . . 455
dynamic-db . . . 456
if-route-exists . . . 457
export (Protocols BGP) . . . 458
export (Protocols IS-IS) . . . 459
export (Protocols DVMRP) . . . 460
export (Protocols LDP) . . . 460
export (Protocols MSDP) . . . 461
export (Protocols OSPF) . . . 462
export (Protocols PIM) . . . 463
export (Protocols PIM Bootstrap) . . . 463
export (Protocols RIP) . . . 464
export (Protocols RIPng) . . . 465
export (Routing Options) . . . 466
import (Protocols BGP) . . . 467
import (Protocols DVMRP) . . . 468
import (Protocols LDP) . . . 469
import (Protocols MSDP) . . . 470
import (Protocols OSPF) . . . 471

import (Protocols PIM) . . . 472
import (Protocols PIM Bootstrap) . . . 473
import (Protocols RIP) . . . 474
import (Protocols RIPng) . . . 475
import (Routing Options) . . . 476
inet (Routing Policy Condition) . . . 476
peer-unit (Routing Policy Condition) . . . 477
policy-options . . . 478
policy-statement . . . 480
prefix-list . . . 484
prefix-list-filter . . . 485
rtf-prefix-list . . . 486
standby (Routing Policy Condition) . . . 487
table . . . 488

Part 3 Administration

Chapter 29 Routing Policy Operational Commands . . . 491

show policy . . . 492
show policy conditions . . . 494
show policy damping . . . 496
show route . . . 498
show route active-path . . . 503
show route advertising-protocol . . . 508
show route all . . . 513
show route aspath-regex . . . 515
show route best . . . 517
show route brief . . . 520
show route community . . . 522
show route community-name . . . 524
show route damping . . . 526
show route detail . . . 531
show route exact . . . 547
show route export . . . 549
show route extensive . . . 552
show route flow validation . . . 568
show route forwarding-table . . . 570
show route hidden . . . 584
show route inactive-path . . . 587
show route inactive-prefix . . . 590
show route instance . . . 592
show route next-hop . . . 599
show route no-community . . . 605
show route output . . . 608
show route protocol . . . 613
show route receive-protocol . . . 625
show route table . . . 633
show route terse . . . 646
show validation database . . . 649

show validation group 651
show validation replication database 653
show validation session 655
show validation statistics 658
test policy 660

Part 4 Index

Index 665

List of Figures

Part 1 Overview

Chapter 1 **Introduction to Routing Policy . . . 3**

Figure 1: Flows of Routing Information and Packets . . . 5
Figure 2: Routing Policies to Control Routing Information Flow . . . 6
Figure 3: Firewall Filters to Control Packet Flow . . . 7
Figure 4: Policy Control Points . . . 7
Figure 5: Importing and Exporting Routes . . . 13

Chapter 2 **Routing Policy Evaluation . . . 17**

Figure 6: Routing Policy Components . . . 17
Figure 7: Routing Policy Evaluation . . . 18

Chapter 3 **Route Filters . . . 25**

Figure 8: Beginning of a Radix Tree . . . 26
Figure 9: First Step of a Radix Tree . . . 26
Figure 10: Second Step of a Radix Tree . . . 26
Figure 11: Locating a Group of Routes . . . 27
Figure 12: Portion of the Radix Tree . . . 31
Figure 13: Route Filter Match Types . . . 32

Chapter 5 **Policy Chains . . . 49**

Figure 14: Routing Policy Chain Evaluation . . . 50

Chapter 6 **Subroutines . . . 51**

Figure 15: Routing Policy Subroutine Evaluation . . . 55

Chapter 12 **Conditional Routing Policies . . . 91**

Figure 16: BGP Import and Export Policies . . . 92

Part 2 Configuration

Chapter 16 **Routing Policy Evaluation . . . 133**

Figure 17: Applying Routing Policies to BGP . . . 135
Figure 18: OSPF with a Conditional Default Route to an ISP . . . 143
Figure 19: ISP Network Example . . . 151
Figure 20: BGP Topology for advertise-peer-as . . . 198
Figure 21: BGP Topology for advertise-external . . . 206
Figure 22: BGP Topology for advertise-inactive . . . 213
Figure 23: BGP Invalid Routes Topology . . . 219
Figure 24: BGP Preference Value Topology . . . 224

Chapter 17 **Route Filters . . . 229**

Figure 25: BGP Topology for Policy Chains 231
Figure 26: Typical Network with IBGP Sessions and Multiple Exit Points 245
Chapter 18 Prefix Lists 261
Figure 27: BGP Topology for Policy Prefix Lists 263
Chapter 19 Subroutines 273
Figure 28: BGP Topology for Policy Subroutine 275
Chapter 20 AS Paths 283
Figure 29: BGP Topology AS Regular Expressions 284
Figure 30: BGP Topology AS Path Prepending 294
Figure 31: Advertisement of Multiple Paths in BGP 302
Chapter 21 Communities 327
Figure 32: Topology for Regular BGP Communities 329
Figure 33: Topology for Extended BGP Communities 343
Figure 34: BGP Policy with a Limit on the Number of Communities Accepted . . 352
Figure 35: BGP Policy That Removes Communities 359
Chapter 22 Testing Policies 367
Figure 36: Routing Policy Test for Complex Regular Expressions 369
Chapter 23 Damp BGP Route Flapping 375
Figure 37: BGP Flap Damping Topology 376
Figure 38: MBGP MVPN with BGP Route Flap Damping 384
Chapter 24 Source Class Usage and Destination Class Usage 395
Figure 39: SCU and DCU Sample Network 396
Chapter 25 Conditional Routing Policies 405
Figure 40: Conditional Installation of Prefixes 408
Chapter 26 Dynamic Routing Policies 421
Figure 41: Dynamic Routing Policy Sample Network 422
Chapter 27 Discard Routing Policy 435
Figure 42: Discard Interface Sample Network 437

List of Tables

About the Documentation xv

Table 1: Notice Icons xvii
Table 2: Text and Syntax Conventions xviii

Part 1 Overview

Chapter 1 Introduction to Routing Policy 3

Table 3: Purpose of Routing Policies and Firewall Filters 8
Table 4: Implementation Differences Between Routing Policies and Firewall Filters 9
Table 5: Default Import and Export Policies for Protocols 15

Chapter 2 Routing Policy Evaluation 17

Table 6: Policy Action Conversion Values 19
Table 7: Policy Expression Logical Operators 19

Chapter 3 Route Filters 25

Table 8: Route Filter Match Types for a Prefix List 29
Table 9: Match Type Examples 32

Chapter 4 Prefix Lists 45

Table 10: Prefix List and Route List Differences 45
Table 11: Route List Match Types for a Prefix List Filter 48

Chapter 7 AS Paths 57

Table 12: AS Path Regular Expression Operators 59
Table 13: Examples of AS Path Regular Expressions 59

Chapter 8 Communities 67

Table 14: Community Attribute Regular Expression Operators 70
Table 15: Examples of Community Attribute Regular Expressions 71

Chapter 10 Damp BGP Route Flapping 81

Table 16: Damping Parameters 81
Table 17: Damping Parameters 83

Chapter 15 Reference Tables 105

Table 18: Match Condition Concepts 106
Table 19: Summary of Key Routing Policy Match Conditions 108
Table 20: Complete List of Routing Policy Match Conditions 109
Table 21: Route List Match Types 116
Table 22: Summary of Key Routing Policy Actions 118
Table 23: Flow Control Actions 121

Table 24: Actions That Manipulate Route Characteristics 121
Table 25: Protocol Support for Import and Export Policies 130

Part 3 Administration

Chapter 29 Routing Policy Operational Commands 491

Table 26: show policy Output Fields 492
Table 27: show policy conditions Output Fields 494
Table 28: show policy damping Output Fields 496
Table 29: show route Output Fields 499
Table 30: show route advertising-protocol Output Fields 509
Table 31: show route damping Output Fields 526
Table 32: show route detail Output Fields 531
Table 33: Next-hop Types Output Field Values 535
Table 34: State Output Field Values 537
Table 35: Communities Output Field Values 539
Table 36: show route export Output Fields 549
Table 37: show route extensive Output Fields 552
Table 38: show route flow validation Output Fields 568
Table 39: show route forwarding-table Output Fields 573
Table 40: show route instance Output Fields 592
Table 41: show route receive-protocol Output Fields 625
Table 42: show route terse Output Fields 646
Table 43: show validation database Output Fields 650
Table 44: show validation group Output Fields 651
Table 45: show validation replication database Output Fields 654
Table 46: show validation session Output Fields 655
Table 47: show validation statistics Output Fields 658

About the Documentation

- Documentation and Release Notes on page xv
- Supported Platforms on page xv
- Using the Examples in This Manual on page xvi
- Documentation Conventions on page xvii
- Documentation Feedback on page xix
- Requesting Technical Support on page xix

Documentation and Release Notes

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

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

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

Supported Platforms

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

- M Series
- T Series
- MX Series
- J Series
- SRX Series
- PTX Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

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

Merging a Full Example

To merge a full example, follow these steps:

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

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

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

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

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

Merging a Snippet

To merge a snippet, follow these steps:

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

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

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

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

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

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

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

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

Documentation Conventions

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

Table 1: Notice Icons

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

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

Table 2: Text and Syntax Conventions

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

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

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

GUI Conventions

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

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

Documentation Feedback

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

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

Requesting Technical Support

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

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

Self-Help Online Tools and Resources

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

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/

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

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

Opening a Case with JTAC

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

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

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

PART 1

Overview

- Introduction to Routing Policy on page 3
- Routing Policy Evaluation on page 17
- Route Filters on page 25
- Prefix Lists on page 45
- Policy Chains on page 49
- Subroutines on page 51
- AS Paths on page 57
- Communities on page 67
- Testing Policies on page 79
- Damp BGP Route Flapping on page 81
- Source Class Usage and Destination Class Usage on page 89
- Conditional Routing Policies on page 91
- Dynamic Routing Policies on page 97
- Discard Routing Policy on page 103
- Reference Tables on page 105

CHAPTER 1

Introduction to Routing Policy

- Policy Framework Overview on page 3
- Comparison of Routing Policies and Firewall Filters on page 8
- Understanding Routing Policies on page 12
- Default Routing Policies on page 15

Policy Framework Overview

The Junos® operating system (Junos OS) provides a *policy framework*, which is a collection of Junos OS policies that allows you to control flows of routing information and packets.

The Junos OS policy architecture is simple and straightforward. However, the actual implementation of each policy adds layers of complexity to the policy as well as adding power and flexibility to your router's capabilities. Configuring a policy has a major impact on the flow of routing information or packets within and through the router. For example, you can configure a routing policy that does not allow routes associated with a particular customer to be placed in the routing table. As a result of this routing policy, the customer routes are not used to forward data packets to various destinations and the routes are not advertised by the routing protocol to neighbors.

Before configuring a policy, determine what you want to accomplish with it and thoroughly understand how to achieve your goal using the various match conditions and actions. Also, make certain that you understand the default policies and actions for the policy you are configuring.

- Routing Policy and Firewall Filters on page 3
- Reasons to Create a Routing Policy on page 4
- Router Flows Affected by Policies on page 4
- Control Points on page 7
- Policy Components on page 8

Routing Policy and Firewall Filters

The policy framework is composed of the following policies:

- Routing policy—Allows you to control the routing information between the routing protocols and the routing tables and between the routing tables and the forwarding

table. All routing protocols use the Junos OS routing tables 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.

- Firewall filter policy—Allows you to control packets transiting the router to a network destination and packets destined for and sent by the router.

NOTE: The term *firewall filter policy* is used here to emphasize that a firewall filter is a policy and shares some fundamental similarities with a routing policy. However, when referring to a firewall filter policy in the rest of this manual, the term *firewall filter* is used.

Reasons to Create a Routing Policy

The following are typical circumstances under which you might want to preempt the default routing policies in the routing policy framework by creating your own routing policies:

- You do not want a protocol to import all routes into the routing table. If the routing table does not learn about certain routes, they can never be used to forward packets and they can never be redistributed into other routing protocols.
- You do not want a routing protocol to export all the active routes it learns.
- You want a routing protocol to announce active routes learned from another routing protocol, which is sometimes called *route redistribution*.
- You want to manipulate route characteristics, such as the preference value, AS path, or community. You can manipulate the route characteristics to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a router's neighbors.
- You want to change the default BGP route flap-damping parameters.
- You want to perform per-packet load balancing.
- You want to enable class of service (CoS).

Router Flows Affected by Policies

The Junos OS policies affect the following router flows:

- Flow of routing information between the routing protocols and the routing tables and between the routing tables and the forwarding table. The Routing Engine handles this flow. *Routing information* is the information about routes learned by the routing protocols from a router's neighbors. This information is stored in routing tables and is subsequently advertised by the routing protocols to the router's neighbors. Routing policies allow you to control the flow of this information.
- Flow of data packets in and out of the router's physical interfaces. The Packet Forwarding Engine handles this flow. *Data packets* are chunks of data that transit the

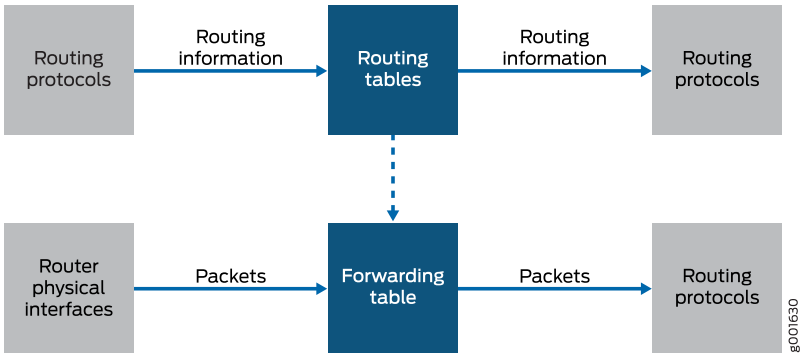
router as they are being forwarded from a source to a destination. When a router receives a data packet on an interface, it determines where to forward the packet by looking in the forwarding table for the best route to a destination. The router then forwards the data packet toward its destination through the appropriate interface. Firewall filters allow you to control the flow of these data packets.

- Flow of local packets from the router's physical interfaces and to the Routing Engine. The Routing Engine handles this flow. *Local packets* are chunks of data that are destined for or sent by the router. Local packets usually contain routing protocol data, data for IP services such as Telnet or SSH, and data for administrative protocols such as the Internet Control Message Protocol (ICMP). When the Routing Engine receives a local packet, it forwards the packet to the appropriate process or to the kernel, which are both part of the Routing Engine, or to the Packet Forwarding Engine. Firewall filters allow you to control the flow of these local packets.

NOTE: **In the rest of this chapter, the term *packets* refers to both data and local packets unless explicitly stated otherwise.**

Figure 1 on page 5 illustrates the flows through the router. Although the flows are very different from each other, they are also interdependent. Routing policies determine which routes are placed in the forwarding table. The forwarding table, in turn, has an integral role in determining the appropriate physical interface through which to forward a packet.

Figure 1: Flows of Routing Information and Packets



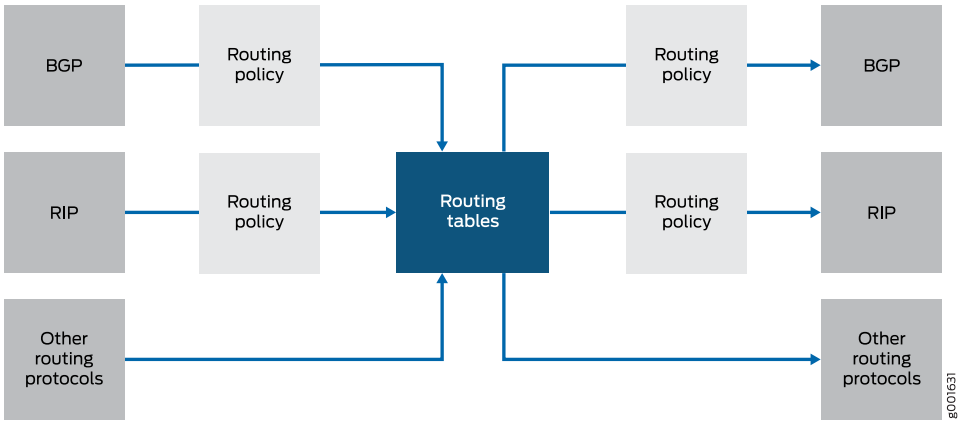
You can configure routing policies to control which routes the routing protocols place in the routing tables and to control which routes the routing protocols advertise from the routing tables (see Figure 2 on page 6). The routing protocols advertise active routes only from the routing tables. (An *active route* is a route that is chosen from all routes in the routing table to reach a destination.)

You can also use routing policies to do the following:

- Change specific route characteristics, which allow you to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a router's neighbors.
- Change to the default BGP route flap-damping values.

- Perform per-packet load balancing.
- Enable class of service (CoS).

Figure 2: Routing Policies to Control Routing Information Flow

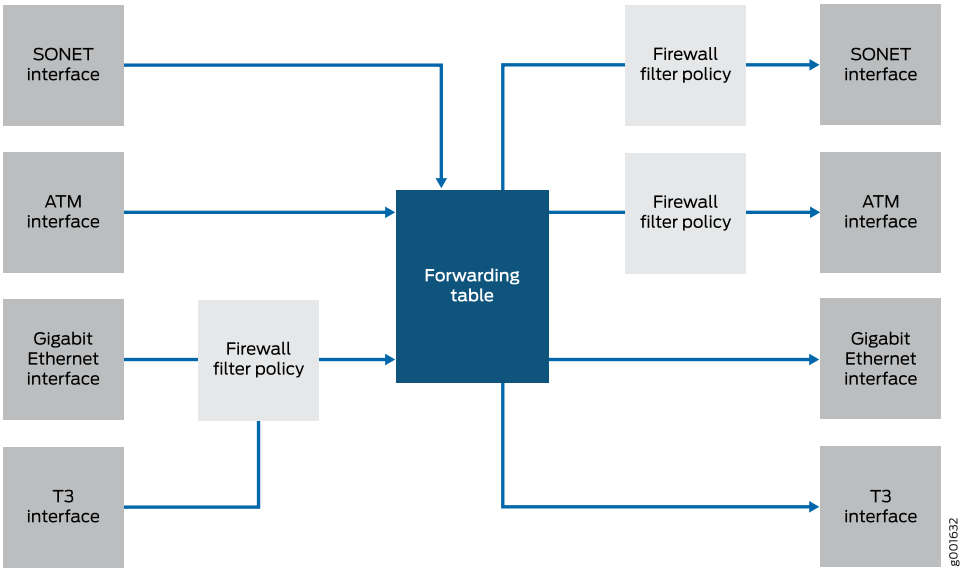


You can configure firewall filters to control the following aspects of packet flow (see Figure 3 on page 7):

- Which data packets are accepted on and transmitted from the physical interfaces. To control the flow of data packets, you apply firewall filters to the physical interfaces.
- Which local packets are transmitted from the physical interfaces and to the Routing Engine. To control local packets, you apply firewall filters on the loopback interface, which is the interface to the Routing Engine.

Firewall filters provide a means of protecting your router from excessive traffic transiting the router to a network destination or destined for the Routing Engine. Firewall filters that control local packets can also protect your router from external incidents such as denial-of-service attacks.

Figure 3: Firewall Filters to Control Packet Flow

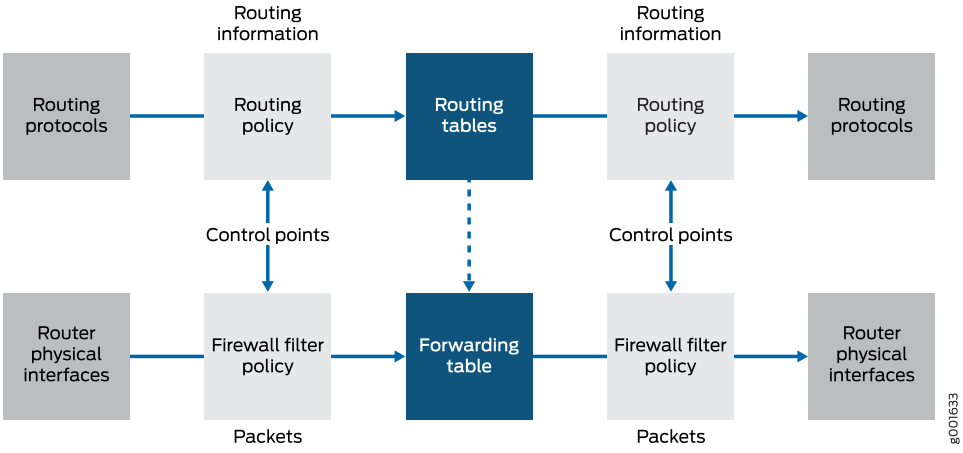


Control Points

All policies provide two points at which you can control routing information or packets through the router (see Figure 4 on page 7). These control points allow you to control the following:

- Routing information before and after it is placed in the routing table.
- Data packets before and after a forwarding table lookup.
- Local packets before and after they are received by the Routing Engine. (Figure 4 on page 7 appears to depict only one control point but because of the bidirectional flow of the local packets, two control points actually exist.)

Figure 4: Policy Control Points



Because there are two control points, you can configure policies that control the routing information or data packets before and after their interaction with their respective tables,

and policies that control local packets before and after their interaction with the Routing Engine. *Import routing policies* control the routing information that is placed in the routing tables, whereas *export routing policies* control the routing information that is advertised from the routing tables. *Input firewall filters* control packets that are received on a router interface, whereas *output firewall filters* control packets that are transmitted from a router interface.

Policy Components

All policies are composed of the following components that you configure:

- *Match conditions*—Criteria against which a route or packets are compared. You can configure one or more criteria. If all criteria match, one or more actions are applied.
- *Actions*—What happens if all criteria match. You can configure one or more actions.
- *Terms*—Named structures in which match conditions and actions are defined. You can define one or more terms.

The policy framework software evaluates each incoming and outgoing route or packet against the match conditions in a term. If the criteria in the match conditions are met, the defined action is taken.

In general, the policy framework software compares the route or packet against the match conditions in the first term in the policy, then goes on to the next term, and so on. Therefore, the order in which you arrange terms in a policy is relevant.

The order of match conditions within a term is not relevant because a route or packet must match all match conditions in a term for an action to be taken.

Related Documentation

- Comparison of Routing Policies and Firewall Filters on page 8
- *Firewall Filters Feature Guide for Routing Devices*
- *Traffic Policers Feature Guide for Routing Devices*

Comparison of Routing Policies and Firewall Filters

Although routing policies and firewall filters share an architecture, their purposes, implementation, and configuration are different. Table 3 on page 8 describes their purposes. Table 4 on page 9 compares the implementation details for routing policies and firewall filters, highlighting the similarities and differences in their configuration.

Table 3: Purpose of Routing Policies and Firewall Filters

Policies	Source	Policy Purpose
Routing policies	Routing information is generated by internal networking peers.	To control the size and content of the routing tables, which routes are advertised, and which routes are considered the best to reach various destinations.

Table 3: Purpose of Routing Policies and Firewall Filters *(continued)*

Policies	Source	Policy Purpose
Firewall filters	Packets are generated by internal and external devices through which hostile attacks can be perpetrated.	To protect your router and network from excessive incoming traffic or hostile attacks that can disrupt network service, and to control which packets are forwarded from which router interfaces.

Table 4: Implementation Differences Between Routing Policies and Firewall Filters

Policy Architecture	Routing Policy Implementation	Firewall Filter Implementation
Control points	Control routing information that is placed in the routing table with an import routing policy and advertised from the routing table with an export routing policy.	Control packets that are accepted on a router interface with an input firewall filter and that are forwarded from an interface with an output firewall filter.
Configuration tasks: • Define policy • Apply policy	Define a policy that contains terms, match conditions, and actions. Apply one or more export or import policies to a routing protocol. You can also apply a *policy expression*, which uses Boolean logical operators with multiple import or export policies. You can also apply one or more export policies to the forwarding table.	Define a policy that contains terms, match conditions, and actions. Apply one input or output firewall filter to a physical interface or physical interface group to filter data packets received by or forwarded to a physical interface (on routing platforms with an Internet Processor II application-specific integrated circuit [ASIC] only). You can also apply one input or output firewall filter to the routing platform's loopback interface, which is the interface to the Routing Engine (on all routing platforms). This allows you to filter local packets received by or forwarded from the Routing Engine.
Terms	Configure as many terms as desired. Define a name for each term. Terms are evaluated in the order in which you specify them. Evaluation of a policy ends after a packet matches the criteria in a term and the defined or default policy action of accept or reject is taken. The route is not evaluated against subsequent terms in the same policy or subsequent policies.	Configure as many terms as desired. Define a name for each term. Terms are evaluated in the order in which you specify them. Evaluation of a firewall filter ends after a packet matches the criteria in a term and the defined or default action is taken. The packet is not evaluated against subsequent terms in the firewall filter.

Table 4: Implementation Differences Between Routing Policies and Firewall Filters *(continued)*

Policy Architecture	Routing Policy Implementation	Firewall Filter Implementation
Match conditions	Specify zero or more criteria that a route must match. You can specify criteria based on source, destination, or properties of a route. You can also specify the following match conditions, which require more configuration: • Autonomous system (AS) path expression—A combination of AS numbers and regular expression operators. • Community—A group of destinations that share a common property. • Prefix list—A named list of prefixes. • Route list—A list of destination prefixes. • Subroutine—A routing policy that is called repeatedly from other routing policies.	Specify zero or more criteria that a packet must match. You must match various fields in the packet's header. The fields are grouped into the following categories: • Numeric values, such as port and protocol numbers. • Prefix values, such as IP source and destination prefixes. • Bit-field values—Whether particular bits in the fields are set, such as IP options, Transmission Control Protocol (TCP) flags, and IP fragmentation fields. You can specify the fields using Boolean logical operators.
Actions	Specify zero or one action to take if a route matches all criteria. You can specify the following actions: • Accept—Accept the route into the routing table, and propagate it. After this action is taken, the evaluation of subsequent terms and policies ends. • Reject—Do not accept the route into the routing table, and do not propagate it. After this action is taken, the evaluation of subsequent terms and policies ends. In addition to the preceding actions, you can also specify zero or more of the following types of actions: • Next term—Evaluate the next term in the routing policy. • Next policy—Evaluate the next routing policy. • Actions that manipulate characteristics associated with a route as the routing protocol places it in the routing table or advertises it from the routing table. • Trace action, which logs route matches.	Specify zero or one action to take if a packet matches all criteria. (We recommend that you always explicitly configure an action.) You can specify the following actions: • Accept—Accept a packet. • Discard—Discard a packet silently, without sending an ICMP message. • Reject—Discard a packet, and send an ICMP destination unreachable message. • Routing instance—Specify a routing table to which packets are forwarded. • Next term—Evaluate the next term in the firewall filter. In addition to zero or the preceding actions, you can also specify zero or more action modifiers. You can specify the following action modifiers: • Count—Add packet to a count total. • Forwarding class—Set the packet forwarding class to a specified value from 0 through 3. • IPsec security association—Used with the source and destination address match conditions, specify an IP Security (IPsec) security association (SA) for the packet. • Log—Store the header information of a packet on the Routing Engine. • Loss priority—Set the packet loss priority (PLP) bit to a specified value, 0 or 1. • Policer—Apply rate-limiting procedures to the traffic. • Sample—Sample the packet traffic. • Syslog—Log an alert for the packet.

Table 4: Implementation Differences Between Routing Policies and Firewall Filters *(continued)*

Policy Architecture	Routing Policy Implementation	Firewall Filter Implementation

Default policies and actions	If an incoming or outgoing route arrives and a policy related to the route is not explicitly configured, the action specified by the default policy for the associated routing protocol is taken.
	
	The following default actions exist for routing policies:
	
	• If a policy does not specify a match condition, all routes evaluated against the policy match.
	• If a match occurs but the policy does not specify an accept, reject, next term, or next policy action, one of the following occurs:
	  ◦ The next term, if present, is evaluated.
	  ◦ If no other terms are present, the next policy is evaluated.
	  ◦ If no other policies are present, the action specified by the default policy is taken.
	• If a match does not occur with a term in a policy and subsequent terms in the same policy exist, the next term is evaluated.
	• If a match does not occur with any terms in a policy and subsequent policies exist, the next policy is evaluated.
	• If a match does not occur by the end of a policy and no other policies exist, the accept or reject action specified by the default policy is taken.
		If an incoming or outgoing packet arrives on an interface and a firewall filter is not configured for the interface, the default policy is taken (the packet is accepted).
		
		The following default actions exist for firewall filters:
		
		• If a firewall filter does not specify a match condition, all packets are considered to match.
		• If a match occurs but the firewall filter does not specify an action, the packet is accepted.
		• If a match occurs, the defined or default action is taken and the evaluation ends. Subsequent terms in the firewall filter are not evaluated, unless the next term action is specified.
		• If a match does not occur with a term in a firewall filter and subsequent terms in the same filter exist, the next term is evaluated.
		• If a match does not occur by the end of a firewall filter, the packet is discarded.

Related Documentation

- Policy Framework Overview on page 3
- *Firewall Filters Feature Guide for Routing Devices*
- *Traffic Policers Feature Guide for Routing Devices*

Understanding Routing Policies

For some routing platform vendors, the flow of routes occurs between various protocols. If, for example, you want to configure redistribution from RIP to OSPF, the RIP process tells the OSPF process that it has routes that might be included for redistribution. In Junos OS, there is not much direct interaction between the routing protocols. Instead, there are central gathering points where all protocols install their routing information. These are the main unicast routing tables inet.0 and inet6.0.

From these tables, the routing protocols calculate the best route to each destination and place these routes in a forwarding table. These routes are then used to forward routing protocol traffic toward a destination, and they can be advertised to neighbors.

- Importing and Exporting Routes on page 12
- Active and Inactive Routes on page 13
- Explicitly Configured Routes on page 14
- Dynamic Database on page 14

Importing and Exporting Routes

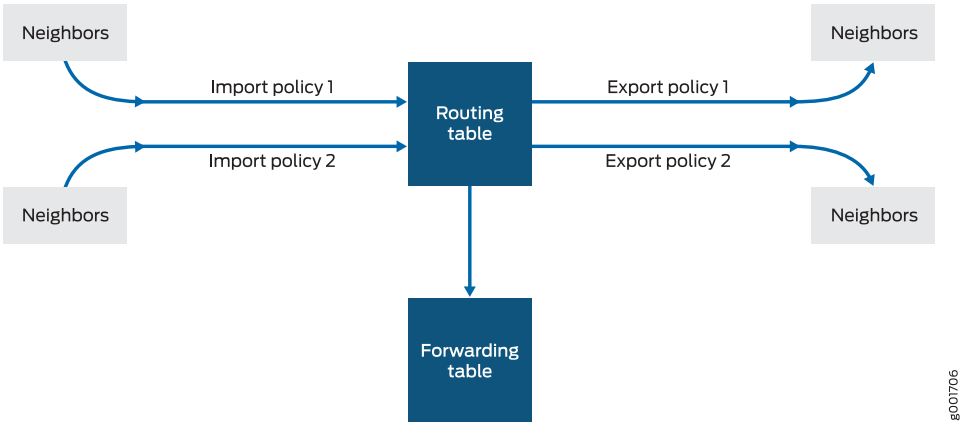
Two terms—*import* and *export*—explain how routes move between the routing protocols and the routing table.

- When the Routing Engine places the routes of a routing protocol into the routing table, it is *importing* routes into the routing table.
- When the Routing Engine uses active routes from the routing table to send a protocol advertisement, it is *exporting* routes from the routing table.

NOTE: The process of moving routes between a routing protocol and the routing table is described always *from the point of view of the routing table*. That is, routes are *imported into* a routing table from a routing protocol and they are *exported from* a routing table to a routing protocol. Remember this distinction when working with routing policies.

As shown in Figure 5 on page 13, you use import routing policies to control which routes are placed in the routing table, and export routing policies to control which routes are advertised from the routing table to neighbors.

Figure 5: Importing and Exporting Routes



In general, the routing protocols place all their routes in the routing table and advertise a limited set of routes from the routing table. The general rules for handling the routing information between the routing protocols and the routing table are known as the *routing policy framework*.

The routing policy framework is composed of default rules for each routing protocol that determine which routes the protocol places in the routing table and advertises from the routing table. The default rules for each routing protocol are known as *default routing policies*.

You can create routing policies to preempt the default policies, which are always present. A *routing policy* allows you to modify the routing policy framework to suit your needs. You can create and implement your own routing policies to do the following:

- Control which routes a routing protocol places in the routing table.
- Control which active routes a routing protocol advertises from the routing table. An *active route* is a route that is chosen from all routes in the routing table to reach a destination.
- Manipulate the route characteristics as a routing protocol places the route in the routing table or advertises the route from the routing table.

You can manipulate the route characteristics to control which route is selected as the active route to reach a destination. The active route is placed in the forwarding table and is used to forward traffic toward the route's destination. In general, the active route is also advertised to a router's neighbors.

Active and Inactive Routes

When multiple routes for a destination exist in the routing table, the protocol selects an active route and that route is placed in the appropriate routing table. For equal-cost routes, the Junos OS places multiple next hops in the appropriate routing table.

When a protocol is exporting routes from the routing table, it exports active routes only. This applies to actions specified by both default and user-defined export policies.

When evaluating routes for export, the Routing Engine uses only active routes from the routing table. For example, if a routing table contains multiple routes to the same destination and one route has a preferable metric, only that route is evaluated. In other words, an export policy does not evaluate all routes; it evaluates only those routes that a routing protocol is allowed to advertise to a neighbor.

NOTE: **By default, BGP advertises active routes. However, you can configure BGP to advertise *inactive routes*, which go to the same destination as other routes but have less preferable metrics.**

Explicitly Configured Routes

An *explicitly configured route* is a route that you have configured. *Direct routes* are not explicitly configured. They are created as a result of IP addresses being configured on an interface Explicitly configured routes include aggregate, generated, local, and static routes. (An *aggregate route* is a route that distills groups of routes with common addresses into one route. A *generated route* is a route used when the routing table has no information about how to reach a particular destination. A *local route* is an IP address assigned to a router interface. A *static route* is an unchanging route to a destination.)

The policy framework software treats direct and explicitly configured routes as if they are learned through routing protocols; therefore, they can be imported into the routing table. Routes cannot be exported from the routing table to the pseudoprotocol, because this protocol is not a real routing protocol. However, aggregate, direct, generated, and static routes can be exported from the routing table to routing protocols, whereas local routes cannot.

Dynamic Database

In Junos OS Release 9.5 and later, you can configure routing policies and certain routing policy objects in a dynamic database that is not subject to the same verification required by the standard configuration database. As a result, you can quickly commit these routing policies and policy objects, which can be referenced and applied in the standard configuration as needed. BGP is the only protocol to which you can apply routing policies that reference policies configured in the dynamic database. After a routing policy based on the dynamic database is configured and committed in the standard configuration, you can quickly make changes to existing routing policies by modifying policy objects in the dynamic database. Because Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.

Related Documentation

- Example: Configuring Dynamic Routing Policies on page 421
- *Example: Redistributing OSPF Routes into IS-IS*

Default Routing Policies

If an incoming or outgoing route or packet arrives and there is no explicitly configured policy related to the route or to the interface upon which the packet arrives, the action specified by the default policy is taken. A *default policy* is a rule or a set of rules that determine whether the route is placed in or advertised from the routing table, or whether the packet is accepted into or transmitted from the router interface.

You must be familiar with the default routing policies to know when you need to modify them to suit your needs. Table 5 on page 15 summarizes the default routing policies for each routing protocol that imports and exports routes. The actions in the default routing policies are taken if you have not explicitly configured a routing policy. This table also shows direct and explicitly configured routes, which for the purposes of this table are considered a pseudoprotocol. Explicitly configured routes include aggregate, generated, and static routes.

Table 5: Default Import and Export Policies for Protocols

Importing or Exporting Protocol	Default Import Policy	Default Export Policy
BGP	Accept all received BGP IPv4 routes learned from configured neighbors and import into the inet.0 routing table. Accept all received BGP IPv6 routes learned from configured neighbors and import into the inet6.0 routing table.	Readvertise all learned BGP routes to all BGP speakers, while following protocol-specific rules that prohibit one IBGP speaker from readvertising routes learned from another IBGP speaker, unless it is functioning as a route reflector.
DVMRP	Accept all DVMRP routes and import into the inet.1 routing table.	Accept and export active DVMRP routes.
IS-IS	Accept all IS-IS routes and import into the inet.0 and inet6.0 routing tables. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)
LDP	Accept all LDP routes and import into the inet.3 routing table.	Reject everything.
MPLS	Accept all MPLS routes and import into the inet.3 routing table.	Accept and export active MPLS routes.
OSPF	Accept all OSPF routes and import into the inet.0 routing table. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)
PIM dense mode	Accept all PIM dense mode routes and import into the inet.1 routing table.	Accept active PIM dense mode routes.

Table 5: Default Import and Export Policies for Protocols *(continued)*

Importing or Exporting Protocol	Default Import Policy	Default Export Policy
PIM sparse mode	Accept all PIM sparse mode routes and import into the inet.1 routing table.	Accept and export active PIM sparse mode routes.

Pseudoprotocol:
• Direct routes
• Explicitly configured routes:
  • Aggregate routes
  • Generated routes
  • Static routes
	Accept all direct and explicitly configured routes and import into the inet.0 routing table.	The pseudoprotocol cannot export any routes from the routing table because it is not a routing protocol.
		
		Routing protocols can export these or any routes from the routing table.

RIP	Accept all RIP routes learned from configured neighbors and import into the inet.0 routing table.	Reject everything. To export RIP routes, you must configure an export policy for RIP.
RIPng	Accept all RIPng routes learned from configured neighbors and import into the inet6.0 routing table.	Reject everything. To export RIPng routes, you must configure an export policy for RIPng.
Test policy	Accept all routes. For additional information about test policy, see "Example: Testing a Routing Policy with Complex Regular Expressions" on page 367.	

You cannot change the default import policy for IS-IS. For OSPF, import policies apply to external routes only. An external route is a route that is outside the OSPF autonomous system (AS). For internal routes (routes learned from OSPF), you cannot change the default import policy for OSPF. As link-state protocols, IS-IS and OSPF exchange routes between systems within an autonomous system (AS). All routers and systems within an AS must share the same link-state database, which includes routes to reachable prefixes and the metrics associated with the prefixes. If an import policy is configured and applied to IS-IS or OSPF, some routes might not be learned or advertised or the metrics for learned routes might be altered, which would make a consistent link-state database impossible.

The default export policy for IS-IS and OSPF protocols is to reject everything. These protocols do not actually export their internally learned routes (the directly connected routes on interfaces that are running the protocol). Both IS-IS and OSPF protocols use a procedure called flooding to announce local routes and any routes learned by the protocol. The flooding procedure is internal to the protocol, and is unaffected by the policy framework. Exporting can be used only to announce information from other protocols, and the default is not to do so.

Related Documentation

- Protocol Support for Import and Export Policies on page 130

CHAPTER 2

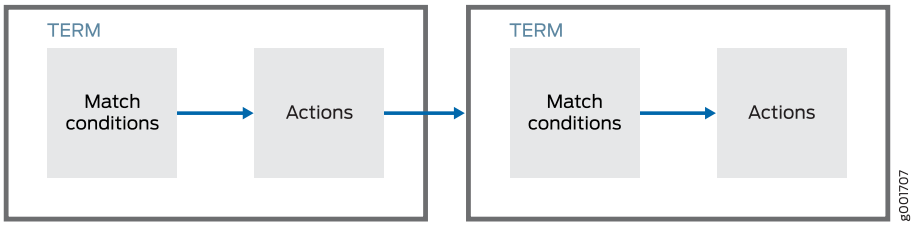
Routing Policy Evaluation

- How a Routing Policy Is Evaluated on page 17
- Understanding Policy Expressions on page 19

How a Routing Policy Is Evaluated

You typically define match conditions and actions within a *term*. Figure 6 on page 17 shows the routing policy components, including the term.

Figure 6: Routing Policy Components



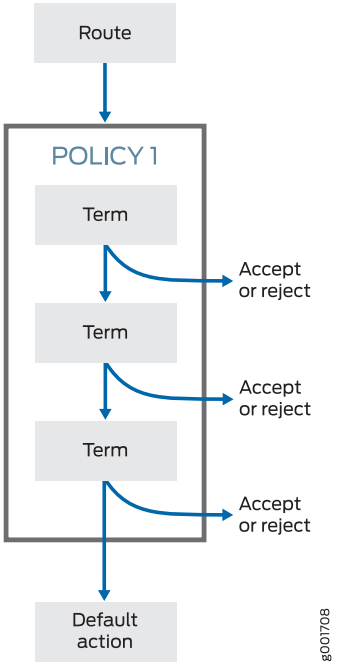
The following actions are taken if the following situations arise during policy evaluation:

- If a policy does not specify a match condition, all routes evaluated against the policy match.
- If a match occurs but the policy does not specify an accept, reject, next term, or next policy action, one of the following occurs:
  - The next term, if present, is evaluated.
  - If no other terms are present, the next policy is evaluated.
  - If no other policies are present, the action specified by the default policy is taken.
- If a match does not occur with a term in a policy and subsequent terms in the same policy exist, the next term is evaluated.
- If a match does not occur with any terms in a policy and subsequent policies exist, the next policy is evaluated.
- If a match does not occur by the end of a policy or all policies, the accept or reject action specified by the default policy is taken.

Figure 7 on page 18 shows how a single routing policy is evaluated. This routing policy consists of multiple terms. Each term consists of match conditions and actions to apply to matching routes. Each route is evaluated against the policy as follows:

1. The route is evaluated against the first term. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the next term action is specified, if no action is specified, or if the route does not match, the evaluation continues as described in Step 2. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.
2. The route is evaluated against the second term. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the next term action is specified, if no action is specified, or if the route does not match, the evaluation continues in a similar manner against the last term. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.
3. If the route matches no terms in the routing policy or the **next policy** action is specified, the accept or reject action specified by the default policy is taken.

Figure 7: Routing Policy Evaluation



Related Documentation

- Default Routing Policies on page 15

Understanding Policy Expressions

Policy expressions give the policy framework software a different way to evaluate routing policies. A *policy expression* uses Boolean logical operators with policies. The logical operators establish rules by which the policies are evaluated.

During evaluation of a routing policy in a policy expression, the policy action of accept, reject, or next policy is converted to the value of TRUE or FALSE. This value is then evaluated against the specified logical operator to produce output of either TRUE or FALSE. The output is then converted back to a flow control action of accept, reject, or next policy. The result of the policy expression is applied as it would be applied to a single policy; the route is accepted or rejected and the evaluation ends, or the next policy is evaluated.

Table 6 on page 19 summarizes the policy actions and their corresponding TRUE and FALSE values and flow control action values. Table 7 on page 19 describes the logical operators. For complete information about policy expression evaluation, see "Policy Expression Evaluation" on page 21.

You must enclose a policy expression in parentheses. You can place a policy expression anywhere in the **import** or **export** statements and in the **from policy** statement.

Table 6: Policy Action Conversion Values

Policy Action	Conversion Value	Flow Control Action Conversion Value
Accept	TRUE	Accept
Reject	FALSE	Reject
Next policy	TRUE	Next policy

Table 7: Policy Expression Logical Operators

Logical Operator	Policy Expression Logic	How Logical Operator Affects Policy Expression Evaluation
&& (Logical AND)	Logical AND requires that all values must be TRUE to produce output of TRUE. Routing policy value of TRUE and TRUE produces output of TRUE. Value of TRUE and FALSE produces output of FALSE. Value of FALSE and FALSE produces output of FALSE.	If the first routing policy returns the value of TRUE, the next policy is evaluated. If the first policy returns the value of FALSE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated.

Table 7: Policy Expression Logical Operators *(continued)*

Logical Operator	Policy Expression Logic	How Logical Operator Affects Policy Expression Evaluation
|| (Logical OR)	Logical OR requires that at least one value must be TRUE to produce output of TRUE. Routing policy value of TRUE and FALSE produces output of TRUE. Value of TRUE and TRUE produces output of TRUE. Value of FALSE and FALSE produces output of FALSE.	If the first routing policy returns the value of TRUE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If the first policy returns the value of FALSE, the next policy is evaluated.
! (Logical NOT)	Logical NOT reverses value of TRUE to FALSE and of FALSE to TRUE. It also reverses the actions of accept and next policy to reject, and reject to accept.	If used with the logical AND operator and the first routing policy value of FALSE is reversed to TRUE, the next policy is evaluated. If the value of TRUE is reversed to FALSE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If used with the logical OR operator and the first routing policy value of FALSE is reversed to TRUE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If the value of TRUE is reversed to FALSE, the next policy is evaluated. If used with a policy and the flow control action is accept or next policy, these actions are reversed to reject. If the flow control action is reject, this action is reversed to accept.

For more information, see the following sections:

- Policy Expression Examples on page 20
- Policy Expression Evaluation on page 21
- Example: Evaluating Policy Expressions on page 22

Policy Expression Examples

The following examples show how to use the logical operators to create policy expressions:

- Logical AND—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is evaluated. If a value of FALSE is returned, **policy2** is not evaluated.

  ```
  export (policy1 && policy2)
  ```

- Logical OR—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is not evaluated. If a value of FALSE is returned, **policy2** is evaluated.

  ```
  export (policy1 || policy2)
  ```

- Logical OR and logical AND—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is skipped and **policy3** is evaluated. If after **policy1** is evaluated, a value of FALSE is returned, **policy2** is evaluated.

If **policy2** returns a value of TRUE, **policy3** is evaluated. If **policy2** returns a value of FALSE, **policy3** is not evaluated.

```
export [(policy1 || policy2) && policy3]
```

- Logical NOT—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, the value is reversed to FALSE and **policy2** is not evaluated. If a value of FALSE is returned, the value is reversed to TRUE and **policy2** is evaluated.

```
export (!policy1 && policy2)
```

The sequential list **[policy1 policy2 policy3]** is not the same as the policy expression **(policy1 && policy2 && policy3)**.

The sequential list is evaluated on the basis of a route matching a routing policy. For example, if **policy1** matches and the action is **accept** or **reject**, **policy2** and **policy3** are not evaluated. If **policy1** does not match, **policy2** is evaluated and so on until a match occurs and the action is **accept** or **reject**.

The policy expressions are evaluated on the basis of the action in a routing policy that is converted to the value of TRUE or FALSE and the logic of the specified logical operator. (For complete information about policy expression evaluation, see "Policy Expression Evaluation" on page 21.) For example, if **policy1** returns a value of FALSE, **policy2** and **policy3** are not evaluated. If **policy1** returns a value of TRUE, **policy2** is evaluated. If **policy2** returns a value of FALSE, **policy3** is not evaluated. If **policy2** returns a value of TRUE, **policy3** is evaluated.

You can also combine policy expressions and sequential lists. In the following example, if **policy1** returns a value of FALSE, **policy2** is evaluated. If **policy2** returns a value of TRUE and contains a **next policy** action, **policy3** is evaluated. If **policy2** returns a value of TRUE but does not contain an action, including a **next policy** action, **policy3** is still evaluated (because if you do not specify an action, next term or next policy are the default actions). If **policy2** returns a value of TRUE and contains an **accept** action, **policy3** is not evaluated.

```
export [(policy1 || policy2) policy3]
```

Policy Expression Evaluation

During evaluation, the policy framework software converts policy actions to values of TRUE or FALSE, which are factors in determining the flow control action that is performed upon a route. However, the software does not actually perform a flow control action on a route until it evaluates an entire policy expression.

The policy framework software evaluates a policy expression as follows:

1. The software evaluates a route against the first routing policy in a policy expression and converts the specified or default action to a value of TRUE or FALSE. (For information about the policy action conversion values, see Table 6 on page 19.)
2. The software takes the value of TRUE or FALSE and evaluates it against the logical operator used in the policy expression (see Table 7 on page 19). Based upon the logical operator used, the software determines whether or not to evaluate the next policy, if one is present.

The policy framework software uses a shortcut method of evaluation: if the result of evaluating a policy predetermines the value of the entire policy expression, the software does not evaluate the subsequent policies in the expression. For example, if the policy expression uses the logical AND operator and the evaluation of a policy returns the value of FALSE, the software does not evaluate subsequent policies in the expression because the final value of the expression is guaranteed to be FALSE no matter what the values of the unevaluated policies.

3. The software performs Step 1 and Step 2 for each subsequent routing policy in the policy expression, if they are present and it is necessary to evaluate them.
4. After evaluating the last routing policy, if it is appropriate, the software evaluates the value of TRUE or FALSE obtained from each routing policy evaluation. Based upon the logical operator used, it calculates an output of TRUE or FALSE.
5. The software converts the output of TRUE or FALSE back to an action. (For information about the policy action conversion values, see Table 6 on page 19.) The action is performed.

   If each policy in the expression returned a value of TRUE, the software converts the output of TRUE back to the flow control action specified in the last policy. For example, if the policy expression **(policy1 && policy2)** is specified and **policy1** specifies **accept** and **policy2** specifies **next term**, the **next term** action is performed.

   If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a policy expression sets a route's metric to 500, this route matches the criteria of **metric 500** defined in the next policy. However, if a route characteristic manipulation action is specified in a policy located in the middle or the end of a policy expression, it is possible, because of the shortcut evaluation, that the policy is never evaluated and the manipulation of the route characteristic never occurs.

Example: Evaluating Policy Expressions

The following sample routing policy uses three policy expressions:

```
[edit]
policy-options {
   policy-statement policy-A {
      from {
         route-filter 10.10.0.0/16 orlonger;
      }
      then reject;
   }
}
policy-options {
   policy-statement policy-B {
      from {
         route-filter 10.20.0.0/16 orlonger;
      }
      then accept;
   }
}
```

```
protocols {
  bgp {
    neighbor 192.168.1.1 {
      export (policy-A && policy-B);
    }
    neighbor 192.168.2.1 {
      export (policy-A || policy-B);
    }
    neighbor 192.168.3.1 {
      export (!policy-A);
    }
  }
}
```

The policy framework software evaluates the transit BGP route 10.10.1.0/24 against the three policy expressions specified in the sample routing policy as follows:

- (policy-A && policy-B)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, and FALSE is evaluated against the specified logical AND. Because the result of FALSE is certain no matter what the results of the evaluation of **policy-B** are (in policy expression logic, any result AND a value of FALSE produces the output of FALSE), **policy-B** is not evaluated and the output of FALSE is produced. The FALSE output is converted to **reject**, and 10.10.1.0/24 is rejected.
- (policy-A || policy-B)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, then FALSE is evaluated against the specified logical OR. Because logical OR requires at least one value of TRUE to produce an output of TRUE, 10.10.1.0/24 is evaluated against **policy-B**. 10.10.1.0/24 does not match **policy-B**, so the default action of **next-policy** is returned. The **next-policy** is converted to a value of TRUE, then the value of FALSE (for **policy-A** evaluation) and TRUE (for **policy-B** evaluation) are evaluated against the specified logical OR. In policy expression logic, FALSE OR TRUE produce an output of TRUE. The output of TRUE is converted to **next-policy**. (TRUE is converted to **next-policy** because **next-policy** was the last action retained by the policy framework software.) **policy-B** is the last routing policy in the policy expression, so the action specified by the default export policy for BGP is taken.
- (!policy-A)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, and FALSE is evaluated against the specified logical NOT. The value of FALSE is reversed to an output of TRUE based on the rules of logical NOT. The output of TRUE is converted to **accept**, and route 10.10.1.0/24 is accepted.

Related Documentation

- Example: Testing a Routing Policy with Complex Regular Expressions on page 367
- Example: Configuring a Policy Subroutine on page 273
- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring Routing Policy Prefix Lists on page 261

CHAPTER 3

Route Filters

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25

Understanding Route Filters for Use in Routing Policy Match Conditions

A *route filter* is a collection of match prefixes. When specifying a match prefix, you can specify an exact match with a particular route or a less precise match. You can configure either a common action that applies to the entire list or an action associated with each prefix.

NOTE: Because the configuration of route filters includes setting up prefixes and prefix lengths, we strongly recommend that you have a thorough understanding of IP addressing, including supernetting, before proceeding with the configuration.

It is also important to understand how a route filter is evaluated, particularly if the route filter includes multiple route-filter options in a from statement. We strongly recommend that you read "How Route Filters Are Evaluated in Routing Policy Match Conditions" on page 33 before proceeding with the configuration. Not fully understanding the evaluation process can result in faulty configuration and unexpected results.

This section discusses the following topics:

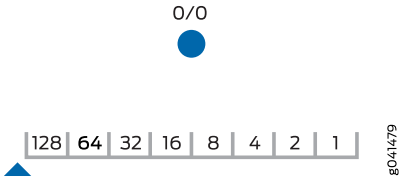
- Radix Trees on page 25
- Configuring Route Filters on page 27
- How Route Filters Are Evaluated in Routing Policy Match Conditions on page 33
- Route Filter Examples on page 35

Radix Trees

To understand the operation of a route filter, you need to be familiar with a device used for binary number matching known as a radix tree (sometimes called a patricia trie or radix trie). A radix tree uses binary lookups to identify IP addresses (routes). Remember that an IP address is a 32-bit number represented in a dotted decimal format for easy comprehension by humans. These 8-bit groupings can each have a value between 0 and 255. A radix tree can be a graphical representation of these binary numbers.

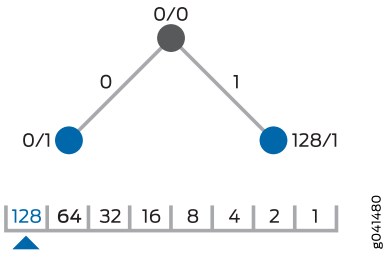
In Figure 8 on page 26. the radix tree starts with no configured value (starts at 0) and is at the leftmost position of the binary IP address. This is shown as 0/0, which is often referred to as the default route.

Figure 8: Beginning of a Radix Tree



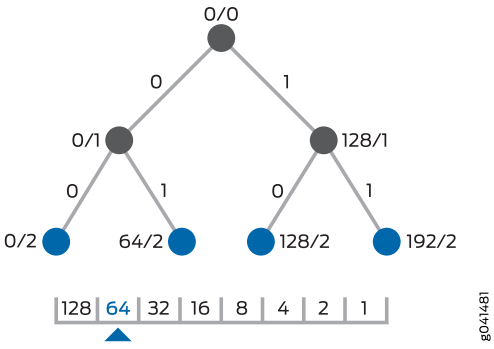
Because this is binary, each bit can have only one of two possible values—a 0 or a 1. Moving down the left branch represents a value of 0, while moving to the right represents a value of 1. The first step is shown in Figure 9 on page 26. At the first position, the first octet of the IP address has a value of 00000000 or 10000000—a 0 or 128, respectively. This is represented in Figure 9 on page 26 by the values 1/1 and 128/1.

Figure 9: First Step of a Radix Tree



The second step is shown in Figure 10 on page 26. This second level of the tree has four possible binary values for the first octet: 00000000, 01000000, 10000000, and 11000000. These decimal values of 0, 64, 128, and 192 are represented by the IP addresses of 0/2, 64/2, 128/2, and 192/2 on the radix tree.

Figure 10: Second Step of a Radix Tree

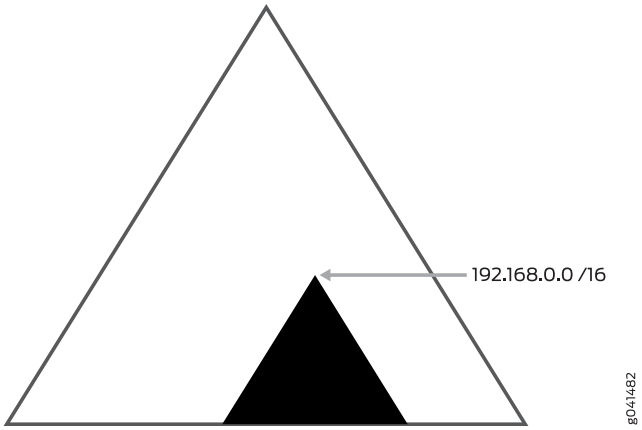


This step-by-step process continues for 33 total levels to represent every possible IP address.

The radix tree structure is helpful when locating a group of routes that all share the same most significant bits. Figure 11 on page 27 shows the point in the radix tree that represents

the 192.168.0.0/16 network. All of the routes that are more specific than 192.168.0.0/16 are shown in the highlighted section.

Figure 11: Locating a Group of Routes



Configuring Route Filters

To configure a route filter, include one or more **route-filter** or **source-address-filter** statements:

```
[edit policy-options policy-statement policy-name term term-name from]
route-filter destination-prefix match-type {
  actions;
}
```

The **route-filter** option is typically used to match an incoming route address to destination match prefixes of any type except for unicast source addresses.

The ***destination-prefix*** address is the IP version 4 (IPv4) or IP version 6 (IPv6) address prefix specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

The **source-address-filter** option is typically used to match an incoming route address to unicast source addresses in multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments.

```
source-address-filter source-prefix match-type {
  actions;
}
```

source-prefix address is the IPv4 or IPv6 address prefix specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32***prefix-length***. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

match-type is the type of match to apply to the source or destination prefix. It can be one of the match types listed in Table 8 on page 29. For examples of the match types and the results when presented with various routes, see Table 9 on page 32.

actions are the actions to take if a route address matches the criteria specified for a destination match prefix (specified as part of a **route-filter** option) or for a source match prefix (specified as part of a **destination-address-filter** option). The actions can consist of one or more of the actions described in "Actions in Routing Policy Terms" on page 119.

In a route filter you can specify actions in two ways:

- In the **route-filter** or **source-address-filter** option—These actions are taken immediately after a match occurs, and the **then** statement is not evaluated.
- In the **then** statement—These actions are taken after a match occurs but no actions are specified for the **route-filter** or **source-address-filter** option.

The **upto** and **prefix-length-range** match types are similar in that both specify the most-significant bits and provide a range of prefix lengths that can match. The difference is that **upto** allows you to specify an upper limit only for the prefix length range, whereas **prefix-length-range** allows you to specify both lower and upper limits.

For more examples of these route filter match types, see "Route Filter Examples" on page 35.

Table 8: Route Filter Match Types for a Prefix List

Match Type	Match Criteria
address-mask ***netmask-value***	All of the following are true: • The bit-wise logical AND of the ***netmask-value*** pattern and the incoming IPv4 or IPv6 route address and the bit-wise logical AND of the ***netmask-value*** pattern and the ***destination-prefix*** address are the same. The bits set in the ***netmask-value*** pattern do not need to be contiguous. • The ***prefix-length*** component of the incoming IPv4 or IPv6 route address and the ***prefix-length*** component of the ***destination-prefix*** address are the same. NOTE: The **address-mask** routing policy match type is valid only for matching an incoming IPv4 (**family inet**) or IPv6 (**family inet6**) route address to a list of destination match prefixes specified in a **route-filter** statement. The **address-mask** routing policy match type enables you to match an incoming IPv4 or IPv6 route address on a configured netmask address in addition to the length of a configured destination match prefix. The length of the route address must match exactly with the length of the configured destination match prefix, as the **address-mask** match type does not support prefix length variations for a range of prefix lengths. When the longest-match lookup is performed on a route filter, the lookup evaluates an **address-mask** match type differently from other routing policy match types. The lookup does not consider the length of the destination match prefix. Instead, the lookup considers the number of contiguous high-order bits set in the netmask value. For more information about this route filter match type, see "How an Address Mask Match Type Is Evaluated" on page 34. For example configurations showing route filters that contain the **address-mask** match type, see the following topics: • "Example: Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix" on page 40. • "Example: Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths" on page 41. • "Example: Evaluation of an Address Mask Match Type with Longest-Match Lookup" on page 42.
exact	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The ***prefix-length*** component of the match prefix is equal to the route's prefix length.
longer	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length is greater than the ***prefix-length*** component of the match prefix.

Table 8: Route Filter Match Types for a Prefix List *(continued)*

Match Type	Match Criteria
orlonger	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or the ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length is equal to or greater than the ***prefix-length*** component of the configured match prefix.
prefix-length-range *prefix-length2-prefix-length3*	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length falls between ***prefix-length2*** and ***prefix-length3***, inclusive.
through {*destination-prefix2* | *source-prefix2*}	All of the following are true: • The route address shares the same most-significant bits as the first match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the first match prefix. • The route address shares the same most-significant bits as the second match prefix (***destination-prefix2*** or ***source-prefix2***). The number of significant bits is described by the ***prefix-length*** component of the second match prefix. • The route's prefix length is less than or equal to the ***prefix-length*** component of the second match prefix. You do not use the **through** match type in most routing policy configurations. For an example, see "Example: Rejecting Routes from Specific Hosts" on page 37.
upto *prefix-length2*	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length falls between the ***prefix-length*** component of the first match prefix and ***prefix-length2***.

Figure 12 on page 31 shows the detailed radix tree for the route 192.168.0.0/16.

Figure 12: Portion of the Radix Tree

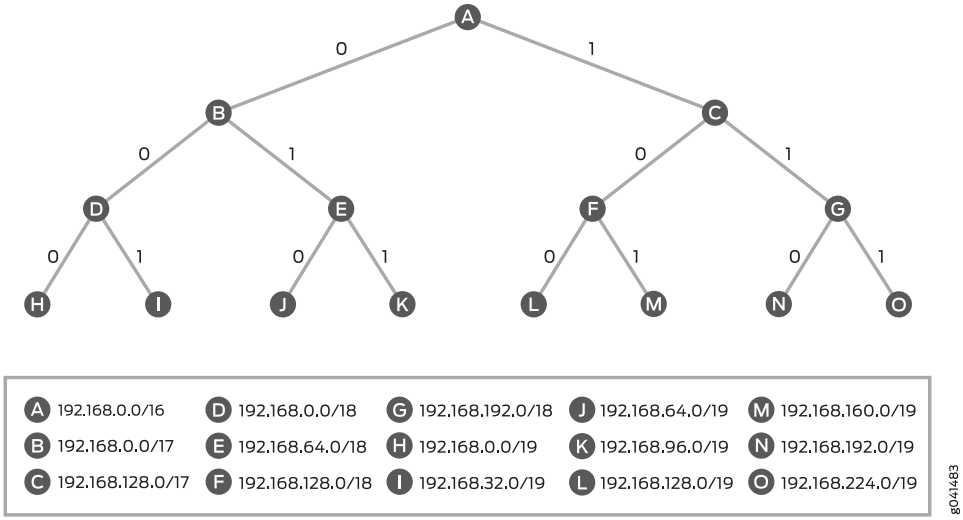


Figure 13 on page 32 and Table 9 on page 32 demonstrate the operation of the various route filter match types.

Figure 13: Route Filter Match Types

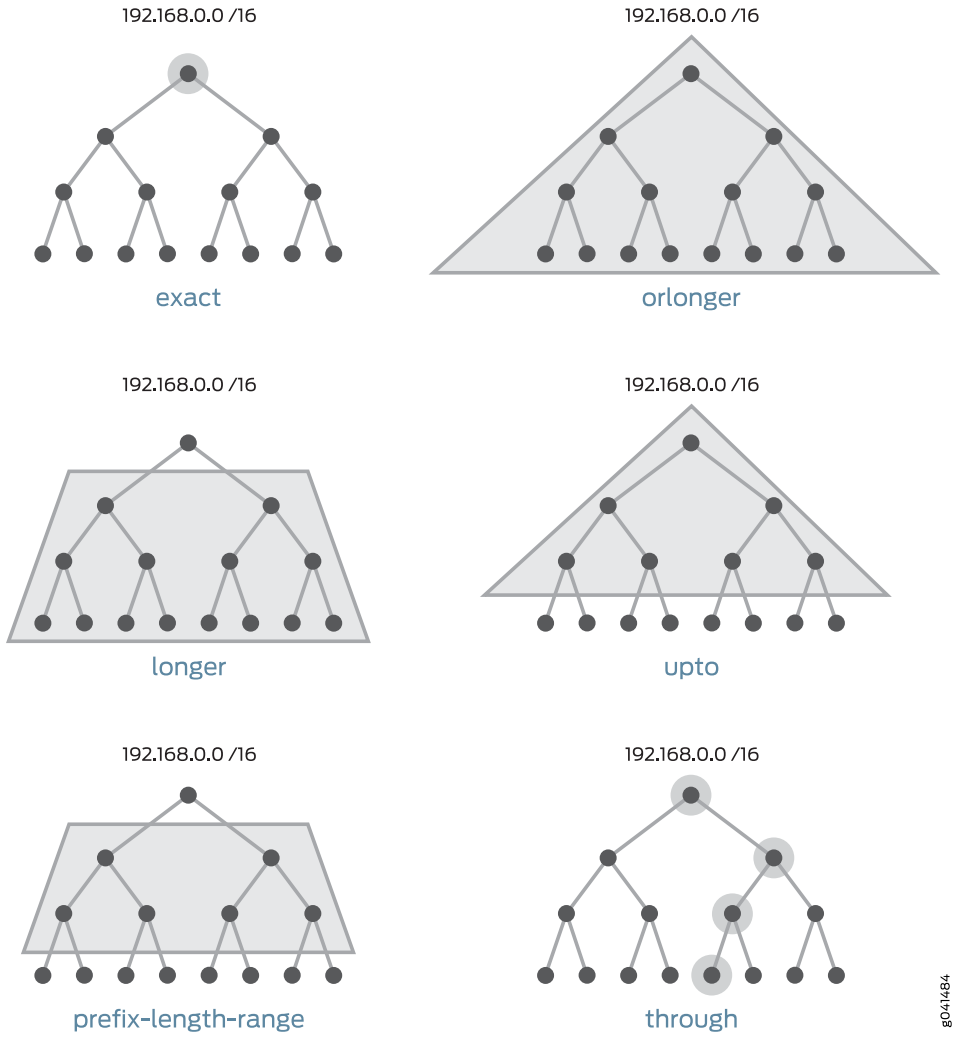


Table 9: Match Type Examples

Prefix	192.168/16 exact	192.168/16 longer	192.168/16 orlonger	192.168/16 upto /24	192.168/16 prefix-length-range /18-/20	192.168/16 through 192.168.16/20	192.168/19 address-mask 255.255.0.0
10.0.0.0/8	–	–	–	–	–	–	–
192.168.0.0/16	Match	–	Match	Match	–	Match	–
192.168.0.0/17	–	Match	Match	Match	–	Match	–
192.168.0.0/18	–	Match	Match	Match	Match	Match	–
192.168.0.0/19	–	Match	Match	Match	Match	Match	Match
192.168.4.0/24	–	Match	Match	Match	–	–	–

Table 9: Match Type Examples *(continued)*

Prefix	192.168/16 exact	192.168/16 longer	192.168/16 orlonger	192.168/16 upto /24	192.168/16 prefix-length-range /18–/20	192.168/16 through 192.168.16/20	192.168/19 address-mask 255.255.0.0
192.168.5.4/30	–	Match	Match	–	–	–	–
192.168.12.4/30	–	Match	Match	–	–	–	–
192.168.12.128/32	–	Match	Match	–	–	–	–
192.168.16.0/20	–	Match	Match	Match	Match	Match	–
192.168.192.0/18	–	Match	Match	Match	Match	–	–
192.168.224.0/19	–	Match	Match	Match	Match	–	Match
10.169.1.0/24	–	–	–	–	–	–	–
10.170.0.0/16	–	–	–	–	–	–	–

How Route Filters Are Evaluated in Routing Policy Match Conditions

During route filter evaluation, the policy framework software compares each route's source address with the destination prefixes in the route filter. The evaluation occurs in two steps:

1. The policy framework software performs a *longest-match lookup*, which means that the software searches for the prefix in the list with the longest length.

   The longest-match lookup considers the ***prefix*** and ***prefix-length*** components of the configured match prefix only, and not the ***match-type*** component. The following sample route filter illustrates this point:

   ```
   from {
     route-filter 192.168.0.0/14 upto /24 reject;
     route-filter 192.168.0.0/15 exact;
   }
   then accept;
   ```

   The longest match for the candidate route 192.168.1.0/24 is the second route-filter, 192.168.0.0/15, which is based on prefix and prefix length only.

2. Once an incoming route matches a prefix (longest first), the following actions occur:

   - The route filter stops evaluating other prefixes, even if the match type fails.
   - The software examines the match type and action associated with that prefix.

NOTE: When a route source address is evaluated against a match criteria that uses the address-mask match type, both steps of the evaluation include the configured netmask value. For more information, see "How an Address Mask Match Type Is Evaluated" on page 34.

In Step 1, if route 192.168.1.0/24 were evaluated, it would fail to match. It matches the longest prefix of 192.168.0.0/15, but it does not match **exact**. The route filter is finished because it matched a prefix, but the result is a failed match because the match type failed.

If a match occurs, the action specified with the prefix is taken. If an action is not specified with the prefix, the action in the **then** statement is taken. If neither action is specified, the software evaluates the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy. For more information about the default routing policies, see "Default Routing Policies" on page 15.

NOTE: If you specify multiple prefixes in the route filter, only one prefix needs to match for a match to occur. The route filter matching is effectively a logical OR operation.

If a match does not occur, the software evaluates the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy.

For example, compare the prefix 192.168.254.0/24 against the following route filter:

```
route-filter 192.168.0.0/16 orlonger;
route-filter 192.168.254.0/23 exact;
```

The prefix 192.168.254.0/23 is determined to be the longest prefix. When the software evaluates 192.168.254.0/24 against the longest prefix, a match occurs (192.168.254.0/24 is a subset of 192.168.254.0/23). Because of the match between 192.168.254.0/24 and the longest prefix, the evaluation continues. However, when the software evaluates the match type, a match does not occur between 192.168.254.0/24 and 192.168.254.0/23 exact. The software concludes that the term does not match and goes on to the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy.

How an Address Mask Match Type Is Evaluated

The **address-mask** routing policy match type enables you to match incoming IPv4 or IPv6 route addresses on a configured netmask value in addition to the length of a configured destination match prefix. During route filter evaluation, an **address-mask** match type is processed differently from other routing policy match types, taking into consideration the configured netmask value:

1. When a longest-match lookup evaluates an **address-mask** routing policy match type, the ***prefix-length*** component of the configured match prefix is not considered. Instead, the lookup considers the number of contiguous high-order bits set in the configured netmask value.
2. When an incoming IPv4 or IPv6 route address is evaluated against a route filter match criteria that uses the **address-mask** routing policy match type, the match succeeds if the following values are identical:
   - The bit-wise logical AND of the configured netmask value and the incoming IPv4 or IPv6 route address

- The bit-wise logical AND of the configured netmask value and the configured destination match prefix

For an example configuration of a route filter that contains two **address-mask** match types, see "Example: Evaluation of an Address Mask Match Type with Longest-Match Lookup" on page 42.

How Prefix Order Affects Route Filter Evaluation

The order in which the prefixes are specified (from top to bottom) typically does not matter, because the policy framework software scans the route filter looking for the longest prefix during evaluation. An exception to this rule is when you use the same destination prefix multiple times in a list. In this case, the order of the prefixes is important, because the list of identical prefixes is scanned from top to bottom, and the first match type that matches the route applies.

In the following example, different match types are specified for the same prefix. The route 0.0.0.0/0 would be rejected, the route 0.0.0.0/8 would be marked with **next-hop self**, and the route 0.0.0.0/25 would be rejected.

```
route-filter 0.0.0.0/0 upto /7 reject;
route-filter 0.0.0.0/0 upto /24 next-hop self;
route-filter 0.0.0.0/0 orlonger reject;
```

Common Configuration Problem with the Longest-Match Lookup

A common problem when defining a route filter is including a shorter prefix that you want to match with a longer, similar prefix in the same list. For example, imagine that the prefix 192.168.254.0/24 is compared against the following route filter:

```
route-filter 192.168.0.0/16 orlonger;
route-filter 192.168.254.0/23 exact;
```

Because the policy framework software performs longest-match lookup, the prefix 192.168.254.0/23 is determined to be the longest prefix. An exact match does not occur between 192.168.254.0/24 and 192.168.254.0/23 exact. The software determines that the term does not match and goes on to the next term or routing policy, if present, or takes the accept or reject action specified by the default policy. (For more information about the default routing policies, see "Default Routing Policies" on page 15.) The shorter prefix 192.168.0.0/16 orlonger that you wanted to match is inadvertently ignored.

One solution to this problem is to remove the prefix 192.168.0.0/16 orlonger from the route filter in this term and move it to another term where it is the only prefix or the longest prefix in the list.

Route Filter Examples

The examples in this section show only fragments of routing policies. Normally, you would combine these fragments with other terms or routing policies.

In all examples, remember that the following actions apply to nonmatching routes:

- Evaluate next term, if present.

- Evaluate next policy, if present.
- Take the accept or reject action specified by the default policy. For more information about the default routing policies, see "Default Routing Policies" on page 15.

The following examples show how to configure route filters for various purposes:

- Example: Rejecting Routes with Specific Destination Prefixes and Mask Lengths on page 36
- Example: Rejecting Routes with a Mask Length Greater than Eight on page 36
- Example: Rejecting Routes with Mask Length Between 26 and 29 on page 37
- Example: Rejecting Routes from Specific Hosts on page 37
- Example: Accepting Routes with a Defined Set of Prefixes on page 38
- Example: Rejecting Routes with a Defined Set of Prefixes on page 38
- Example: Rejecting Routes with Prefixes Longer than 24 Bits on page 38
- Example: Rejecting PIM Multicast Traffic Joins on page 39
- Example: Rejecting PIM Traffic on page 39
- Example: Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix on page 40
- Example: Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths on page 41
- Example: Evaluation of an Address Mask Match Type with Longest-Match Lookup on page 42

Example: Rejecting Routes with Specific Destination Prefixes and Mask Lengths

Reject routes with a destination prefix of 0.0.0.0 and a mask length from 0 through 8, and accept all other routes:

```
[edit]
policy-options {
   policy-statement policy-statement from-hall2 {
      term 1 {
         from {
            route-filter 0.0.0.0/0 upto /8 reject;
         }
      }
      then accept;
   }
}
```

Example: Rejecting Routes with a Mask Length Greater than Eight

Reject routes with a mask of /8 and greater (that is, /8, /9, /10, and so on) that have the first 8 bits set to 0 and accept routes less than 8 bits in length:

```
[edit]
policy-options {
   policy-statement from-hall3 {
      term term1 {
```

```
        from {
          route-filter 0/0 upto /7 accept;
          route-filter 0/8 orlonger;
        }
        then reject;
      }
    }
  }
```

Example: Rejecting Routes with Mask Length Between 26 and 29

Reject routes with the destination prefix of 192.168.10/24 and a mask between /26 and /29 and accept all other routes:

```
[edit]
policy-options {
  policy-statement from-customer-a {
    term term1 {
      from {
        route-filter 192.168.10/24 prefix-length-range /26–/29 reject;
      }
      then accept;
    }
  }
}
```

Example: Rejecting Routes from Specific Hosts

Reject a range of routes from specific hosts, and accept all other routes:

```
[edit]
policy-options {
  policy-statement hosts-only {
    from {
      route-filter 10.125.0.0/16 upto /31 reject;
      route-filter 0/0;
    }
    then accept;
  }
}
```

You do not use the **through** match type in most routing policy configurations. You should think of **through** as a tool to group a contiguous set of exact matches. For example, instead of specifying four exact matches:

```
from route-filter 0.0.0.0/1 exact
from route-filter 0.0.0.0/2 exact
from route-filter 0.0.0.0/3 exact
from route-filter 0.0.0.0/4 exact
```

You could represent them with the following single match:

```
from route-filter 0.0.0.0/1 through 0.0.0.0/4
```

Example: Accepting Routes with a Defined Set of Prefixes

Explicitly accept a limited set of prefixes (in the first term) and reject all others (in the second term):

```
policy-options {
policy-statement internet-in {
   term 1 {
      from {
         route-filter 192.168.231.0/24 exact accept;
         route-filter 192.168.244.0/24 exact accept;
         route-filter 192.168.198.0/24 exact accept;
         route-filter 192.168.160.0/24 exact accept;
         route-filter 192.168.59.0/24 exact accept;
      }
   }
   term 2 {
      then {
         reject;
      }
   }
}
```

Example: Rejecting Routes with a Defined Set of Prefixes

Reject a few groups of prefixes, and accept the remaining prefixes:

```
[edit policy-options]
policy-statement drop-routes {
   term 1{
      from { # first, reject a number of prefixes:
         route-filter default exact reject; # reject 0.0.0.0/0 exact
         route-filter 0.0.0.0/8 orlonger reject; # reject prefix 0, mask /8 or longer
         route-filter 10.0.0.0/8 orlonger reject; # reject loopback addresses
         }
         route-filter 10.105.0.0/16 exact { # accept 10.105.0.0/16
            as-path-prepend “1 2 3”;
            accept;
         }
         route-filter 192.0.2.0/24 orlonger reject; # reject test network packets
         route-filter 224.0.0.0/3 orlonger reject; # reject multicast and higher
         route-filter 0.0.0.0/0 upto /24 accept; # accept everything up to /24
         route-filter 0.0.0.0/0 orlonger accept; # accept everything else
         }
      }
   }
}
```

Example: Rejecting Routes with Prefixes Longer than 24 Bits

Reject all prefixes longer than 24 bits. You would install this routing policy in a sequence of routing policies in an **export** statement. The first term in this filter passes on all routes with a prefix length of up to 24 bits. The second, unnamed term rejects everything else.

```
[edit policy-options]
policy-statement 24bit-filter {
```

```
    term acl20 {
      from {
        route-filter 0.0.0.0/0 upto /24;
      }
      then next policy;
    }
    then reject;
  }
```

If, in this example, you were to specify **route-filter 0.0.0.0/0 upto /24 accept**, matching prefixes would be accepted immediately and the next routing policy in the **export** statement would never get evaluated.

If you were to include the **then reject** statement in the term **acl20**, prefixes greater than 24 bits would never get rejected because the policy framework software, when evaluating the term, would move on to evaluating the next statement before reaching the **then reject** statement.

Example: Rejecting PIM Multicast Traffic Joins

Configure a routing policy for rejecting Protocol Independent Multicast (PIM) multicast traffic joins for a source destination prefix from a neighbor:

```
[edit]
policy-options {
  policy-statement join-filter {
    from {
      neighbor 10.14.12.20;
      source-address-filter 10.83.0.0/16 orlonger;
    }
    then reject;
  }
}
```

Example: Rejecting PIM Traffic

Configure a routing policy for rejecting PIM traffic for a source destination prefix from an interface:

```
[edit]
policy-options {
  policy-statement join-filter {
    from {
      interface so-1/0/0.0;
      source-address-filter 10.83.0.0/16 orlonger;
    }
    then reject;
  }
}
```

The following routing policy qualifiers apply to PIM:

- **interface**—Interface over which a join is received
- **neighbor**—Source from which a join originates

- **route-filter**—Group address
- **source-address-filter**—Source address for which to reject a join

For more information about importing a PIM join filter in a PIM protocol definition, see the *Multicast Protocols Feature Guide for Routing Devices*.

Example: Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix

Accept incoming IPv4 routes with a destination prefix of 10.1.0/24 and the third byte an even number from 0 to 14, inclusive:

```
[edit]
policy-options {
  policy-statement from_customer_a {
    term term_1 {
      from {
        route-filter 10.1.0.0/24 address-mask 255.255.241.0;
      }
      then {
        ...
        reject;
      }
    }
  }
}
```

The route filter in routing policy term **term_1** matches the following incoming IPv4 route addresses:

- 10.1.0.0/24
- 10.1.2.0/24
- 10.1.4.0/24
- 10.1.6.0/24
- 10.1.8.0/24
- 10.1.10.0/24
- 10.1.12.0/24
- 10.1.14.0/24

The bit-wise logical AND of the netmask value and the candidate route address must match the bit-wise logical AND of the netmask value and the match prefix address. That is, where the netmask bit pattern 255.255.241.0 contains a set bit, the incoming IPv4 route address being evaluated must match the value of the corresponding bit in the destination prefix address 10.1.0.0/24.

- The first two bytes of the netmask value are binary 1111 1111 1111 1111, which means that a candidate route address will fail the match if the first two bytes are not 10.1.

- The third byte of the netmask value is binary 1111 0001, which means that a candidate route address will fail the match if the third byte is greater than 15 (decimal), an odd number, or both.
- The prefix length of the match prefix address is 24 (decimal), which means that a candidate route address will fail the match if its prefix length is not exactly 24.

As an example, suppose that the candidate route address being tested in the policy is 10.1.8.0/24 (binary 0000 1010 0000 0001 0000 1000).

1. When the netmask value is applied to this candidate route address, the result is binary 0000 1010 0000 0001 0000 0000.
2. When the netmask value is applied to the configured destination prefix address, the result is also binary 0000 1010 0000 0001 0000 0000.
3. Because the results of both AND operations are the same, the match continues to the second match criteria.
4. Because the prefix lengths of the candidate address and the configured destination prefix address are the same (24 bits), the match succeeds.

As another example, suppose that the candidate route address being tested in the policy is 10.1.3.0/24 (binary 0000 1010 0000 0001 0000 0011).

1. When the netmask value is applied to this candidate route address, the result is binary 0000 1010 0000 0001 0000 0001.
2. However, when the netmask value is applied to the configured destination prefix address, the result is binary 0000 1010 0000 0001 0000 0000.
3. Because the results of the two AND operations are different (in the third byte), the match fails.

Example: Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths

Accept incoming IPv4 route addresses of the form 10.*.1/24 or 10.*.1.*/32:

```
[edit]
policy-options {
   policy-statement from_customer_b {
      term term_2 {
         from {
            route-filter 10.0.1.0/24 address-mask 255.0.255.0;
            route-filter 10.0.1.0/32 address-mask 255.0.255.0;
         }
         then {
            ...
            reject;
         }
      }
   }
}
```

The route filter match criteria **10.0.1.0/24 address-mask 255.0.255.0** matches an incoming IPv4 route address of the form 10.*.1/24. The route's prefix length must be exactly 24 bits long, and any value is acceptable in the second byte.

The route filter match criteria **10.0.1.0/32 address-mask 255.0.255.0** matches an incoming IPv4 route address of the form 10.*.1.*/32. The route's prefix length must be exactly 32 bits long, and any value is acceptable in the second byte and the fourth byte.

Example: Evaluation of an Address Mask Match Type with Longest-Match Lookup

This example illustrates how a longest-match lookup evaluates a route filter that contains two **address-mask** match types. Consider the route filter configured in the routing policy term **term_3** below:

```
[edit]
policy-options {
   policy-statement from_customer_c {
      term term_3 {
         from {
            route-filter 10.0.1.0/24 address-mask 255.0.255.0;
            route-filter 10.0.2.0/24 address-mask 255.240.255.0;
         }
         then {
            ...
         }
      }
   }
}
```

Suppose that the incoming IPv4 route source address 10.1.1.0/24 is tested against the route filter configured in the policy term **term_3**:

1. The longest-match lookup tree for routing policy term **term_3** contains two match prefixes: one prefix for **10.0.1.0/24 address-mask 255.0.255.0** and one prefix for **10.0.2.0/24 address-mask 255.240.255.0**. When searching the tree for the longest-prefix match for a candidate, the longest-match lookup considers the number of contiguous high-order bits in the configured ***netmask-value*** instead of the length of the configured ***destination-prefix***:

   - For the first route filter match criteria, the longest-match lookup entry is 10.0.0.0/8 because the netmask value contains 8 contiguous high-order bits.
   - For second route filter match criteria, the longest-match lookup entry is 10.0.0.0/12 because the netmask value contains 12 contiguous high-order bits.

   For the candidate route address 10.1.1.0/24, the longest-match lookup returns the tree entry 10.0.0.0/12, which is corresponds to the route filter match criteria **10.0.2.0/24 address-mask 255.240.255.0**.

2. Now that the longest-match prefix in **term_3** has been identified for the candidate route address, the candidate route address is evaluated against the route filter match criteria **10.0.2.0/24 address-mask 255.240.255.0**:

   a. To test the incoming IPv4 route address 10.1.1.0/24, the netmask value 255.240.255.0 is applied to 10.1.1.0/24. The result is 10.0.1.0.

b. To test the configured destination prefix address 10.0.2.0/24, the netmask value 255.240.255.0 is applied to 10.0.2.0/24. The result is 10.0.2.0.

c. Because the results are different, the route filter match fails. No actions, whether specified with the match criteria or with the **then** statement, are taken. The incoming IPv4 route address is not evaluated against any other match criteria.

Related Documentation

- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF on page 240
- Example: Configuring the MED Using Route Filters on page 245

CHAPTER 4

Prefix Lists

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 45

Understanding Prefix Lists for Use in Routing Policy Match Conditions

A *prefix list* is a named list of IP addresses. You can specify an exact match with incoming routes and apply a common action to all matching prefixes in the list.

Suppose, for example, that you configure the following prefix list:

```
prefix-list bgp179 {
  apply-path "protocols bgp group <*> neighbor <*>";
}
```

This works well when all neighbors on the device are in the same address family.

When the neighbors are in different address families, for example when both IPv4 and IPv6 neighbors are configured, you can use a prefix list as follows:

```
prefix-list IPV4-BGP-NEIGHBORS {
  apply-path "protocols bgp group <*> neighbor <*.*.*.*>";
}
prefix-list IPV6-BGP-NEIGHBORS {
  apply-path "protocols bgp group <*> neighbor <*:*:*:*>";
}
```

One prefix list matches IPv4 addresses. The other matches IPv6 addresses. You can run the **show configuration policy-options prefix-list prefix-list name | display inheritance** command to verify the configuration.

A prefix list functions like a route list that contains multiple instances of the **exact** match type only. The differences between these two extended match conditions are summarized in Table 10 on page 45.

Table 10: Prefix List and Route List Differences

Feature	Prefix List	Route Lists
Action	Can specify action in a **then** statement only. These actions are applied to all prefixes that match the term.	Can specify action that is applied to a particular prefix in a **route-filter** match condition in a **from** statement, or to all prefixes in the list using a **then** statement.

For information about configuring route lists, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 25.

This section includes the following information:

- Configuring Prefix Lists on page 46
- How Prefix Lists Are Evaluated in Routing Policy Match Conditions on page 47
- Configuring Prefix List Filters on page 47

Configuring Prefix Lists

You can create a named prefix list and include it in a routing policy with the **prefix-list** match condition (described in "Routing Policy Match Conditions" on page 107).

To define a prefix list, include the **prefix-list** statement:

```
[edit policy-options]
  prefix-list prefix-list-name {
  apply-path path;
  ip-addresses;
}
```

You can use the **apply-path** statement to include all prefixes pointed to by a defined path, or you can specify one or more addresses, or both.

To include a prefix list in a routing policy, specify the **prefix-list** match condition in the **from** statement at the **[edit policy-options policy-statement *policy-name* term *term-name*]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name]
from {
  prefix-list prefix-list-name;
}
then actions;
```

name identifies the prefix list. It 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 quotation marks (" ").

ip-addresses are the IPv4 or IP version 6 (IPv6) prefixes specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32***prefix-length***. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

NOTE: **You cannot apply actions to individual prefixes in the list.**

You can specify the same prefix list in the **from** statement of multiple routing policies or firewall filters. For information about firewall filters, see *Guidelines for Configuring Firewall Filters* and *Guidelines for Applying Firewall Filters*.

Use the **apply-path** statement to configure a prefix list comprising all IP prefixes pointed to by a defined path. This eliminates most of the effort required to maintain a group prefix list.

The path consists of elements separated by spaces. Each element matches a configuration keyword or an identifier, and you can use wildcards to match more than one identifier. Wildcards must be enclosed in angle brackets, for example, <*>.

NOTE: **You cannot add a path element, including wildcards, after a leaf statement in the apply-path statement. Path elements, including wildcards, can only be used after a container statement.**

NOTE: **When you use apply-path to define a prefix list, you can also use the same prefix list in a policy statement.**

For examples of configuring a prefix list, see "Example: Configuring Routing Policy Prefix Lists" on page 261.

How Prefix Lists Are Evaluated in Routing Policy Match Conditions

During prefix list evaluation, the policy framework software performs a *longest-match lookup*, which means that the software searches for the prefix in the list with the longest length. The order in which you specify the prefixes, from top to bottom, does not matter. The software then compares a route's source address to the longest prefix.

You can use prefix list qualifiers for prefixes contained in a prefix list by configuring a prefix list filter. For more information, see *Configuring Prefix Lists for Use in Routing Policy Match Conditions*.

If a match occurs, the evaluation of the current term continues. If a match does not occur, the evaluation of the current term ends.

NOTE: **If you specify multiple prefixes in the prefix list, only one prefix must match for a match to occur. The prefix list matching is effectively a logical OR operation.**

Configuring Prefix List Filters

A prefix list filter allows you to apply prefix list qualifiers to a list of prefixes within a prefix list. The prefixes within the list are evaluated using the specified qualifiers. You can configure multiple prefix list filters under the same policy term.

To configure a prefix list filter, include the **prefix-list-filter** statement at the **[edit policy-options policy-statement *policy-name* from]** hierarchy level:

```
[edit policy-options policy-statement policy-name]
from {
```

```
    prefix-list-filter prefix-list-name match-type actions;
}
```

The ***prefix-list-name*** option is the name of the prefix list to be used for evaluation. You can specify only one prefix list.

The ***match-type*** option is the type of match to apply to the prefixes in the prefix list. It can be one of the match types listed in Table 11 on page 48.

The ***actions*** option is the action to take if the prefix list matches. It can be one or more of the actions listed in "Configuring Flow Control Actions" on page 120 and "Configuring Actions That Manipulate Route Characteristics" on page 121.

Table 11: Route List Match Types for a Prefix List Filter

Match Type	Match Condition
exact	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to the route's prefix length.
longer	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is greater than the route's prefix length.
orlonger	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to or greater than the route's prefix length.

Related Documentation

- Example: Configuring Routing Policy Prefix Lists on page 261
- *Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List*

CHAPTER 5

Policy Chains

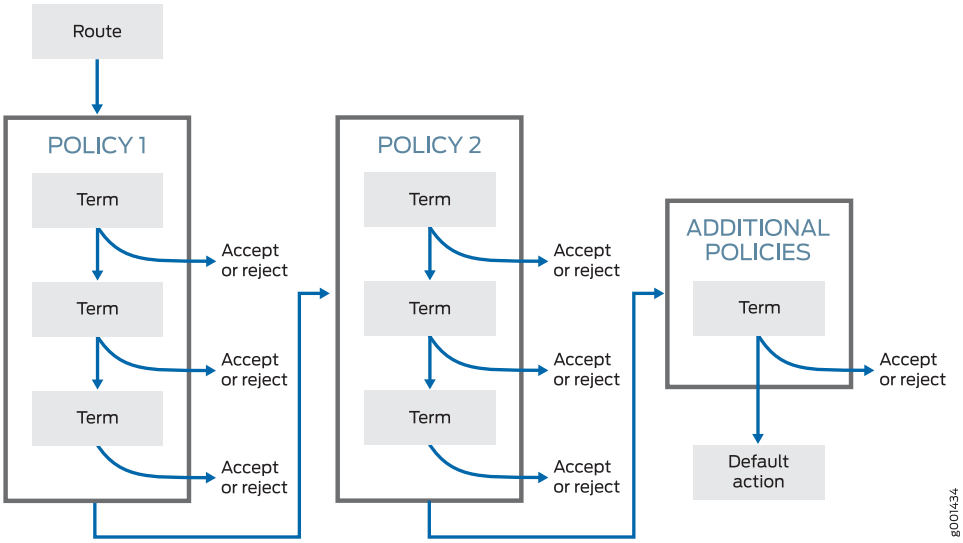
- How a Routing Policy Chain Is Evaluated on page 49

How a Routing Policy Chain Is Evaluated

Figure 14 on page 50 shows how a chain of routing policies is evaluated. These routing policies consist of multiple terms. Each term consists of match conditions and actions to apply to matching routes. Each route is evaluated against the policies as follows:

1. The route is evaluated against the first term in the first routing policy. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the **next term** action is specified, if no action is specified, or if the route does not match, the evaluation continues as described in Step 2. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.
2. The route is evaluated against the second term in the first routing policy. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the **next term** action is specified, if no action is specified, or if the route does not match, the evaluation continues in a similar manner against the last term in the first routing policy. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.
3. If the route does not match a term or matches a term with a **next policy** action in the first routing policy, it is evaluated against the first term in the second routing policy.
4. The evaluation continues until the route matches a term with an accept or reject action defined or until there are no more routing policies to evaluate. If there are no more routing policies, then the accept or reject action specified by the default policy is taken.

Figure 14: Routing Policy Chain Evaluation



Related Documentation

- Default Routing Policies on page 15
- Example: Configuring Policy Chains and Route Filters on page 229

CHAPTER 6

Subroutines

- Understanding Policy Subroutines in Routing Policy Match Conditions on page 51
- How a Routing Policy Subroutine Is Evaluated on page 54

Understanding Policy Subroutines in Routing Policy Match Conditions

You can use a routing policy called from another routing policy as a match condition. This process makes the called policy a *subroutine*.

In some ways, the Junos OS policy framework is similar to a programming language. This similarity includes the concept of nesting policies into a policy subroutine. A subroutine in a software program is a section of code that you reference on a regular basis. A policy subroutine works in the same fashion—you reference an existing policy as a match criterion in another policy. The routing device first evaluates the subroutine and then evaluates the main policy. The evaluation of the subroutine returns a true or false Boolean result to the main policy. Because you are referencing the subroutine as a match criterion, a true result means that the main policy has a match and can perform any configured actions. A false result from the subroutine, however, means that the main policy does not have a match.

Configuring Subroutines

To configure a subroutine in a routing policy to be called from another routing policy, create the subroutine and specify its name using the **policy** match condition in the **from** or **to** statement of another routing policy.

NOTE: **Do not evaluate a routing policy within itself. The result is that no prefixes ever match the routing policy.**

The action specified in a subroutine is used to provide a match condition to the calling policy. If the subroutine specifies an action of accept, the calling policy considers the route to be a match. If the subroutine specifies an action of reject, the calling policy considers the route not to match. If the subroutine specifies an action that is meant to manipulate the route characteristics, the changes are made.

Possible Consequences of Termination Actions in Subroutines

A subroutine with particular statements can behave differently from a routing policy that contains the same statements. With a subroutine, you must remember that the possible termination actions of accept or reject specified by the subroutine or the default policy can greatly affect the expected results.

In particular, you must consider what happens if a match does not occur with routes specified in a subroutine and if the default policy action that is taken is the action that you expect and want.

For example, imagine that you are a network administrator at an Internet service provider (ISP) that provides service to Customer A. You have configured several routing policies for the different classes of neighbors that Customer A presents on various links. To save time maintaining the routing policies for Customer A, you have configured a subroutine that identifies their routes and various routing policies that call the subroutine, as shown below:

```
[edit]
policy-options {
   policy-statement customer-a-subroutine {
      from {
         route-filter 10.1/16 exact;
         route-filter 10.5/16 exact;
         route-filter 192.168.10/24 exact;
      }
      then accept;
   }
}
policy-options {
   policy-statement send-customer-a-default {
      from {
         policy customer-a-subroutine;
      }
      then {
         set metric 500;
         accept;
      }
   }
}
policy-options {
   policy-statement send-customer-a-primary {
      from {
         policy customer-a-subroutine;
      }
      then {
         set metric 100;
         accept;
      }
   }
}
policy-options {
   policy-statement send-customer-a-secondary {
      from {
```

```
        policy customer-a-subroutine;
      }
      then {
        set metric 200;
        accept;
      }
    }
  }
  protocols {
    bgp {
      group customer-a {
        export send-customer-a-default;
        neighbor 10.1.1.1;
        neighbor 10.1.2.1;
        neighbor 10.1.3.1 {
          export send-customer-a-primary;
        }
        neighbor 10.1.4.1 {
          export send-customer-a-secondary;
        }
      }
    }
  }
```

The following results occur with this configuration:

- The group-level **export** statement resets the metric to 500 when advertising all BGP routes to neighbors 10.1.1.1 and 10.1.2.1 rather than just the routes that match the subroutine route filters.
- The neighbor-level **export** statements reset the metric to 100 and 200 when advertising all BGP routes to neighbors 10.1.3.1 and 10.1.4.1, respectively, rather than just the BGP routes that match the subroutine route filters.

These unexpected results occur because the subroutine policy does not specify a termination action for routes that do not match the route filter and therefore, the default BGP export policy of accepting all BGP routes is taken.

If the statements included in this particular subroutine had been contained within the calling policies themselves, only the desired routes would have their metrics reset.

This example illustrates the differences between routing policies and subroutines and the importance of the termination action in a subroutine. Here, the default BGP export policy action for the subroutine was not carefully considered. A solution to this particular example is to add one more term to the subroutine that rejects all other routes that do not match the route filters:

```
[edit]
policy-options {
  policy-statement customer-a-subroutine {
    term accept-exact {
      from {
        route-filter 10.1/16 exact;
        route-filter 10.5/16 exact;
        route-filter 192.168.10/24 exact;
```

```
        }
        then accept;
      }
      term reject-others {
        then reject;
      }
    }
  }
```

Termination action strategies for subroutines in general include the following:

- Depend upon the default policy action to handle all other routes.
- Add a term that accepts all other routes.
- Add a term that rejects all other routes.

The option that you choose depends upon what you want to achieve with your subroutine. Plan your subroutines carefully.

Related Documentation

- How a Routing Policy Subroutine Is Evaluated on page 54
- Example: Configuring a Policy Subroutine on page 273

How a Routing Policy Subroutine Is Evaluated

Figure 15 on page 55 shows how a subroutine is evaluated. The subroutine is included in the first term of the first routing policy in a chain. Each route is evaluated against the subroutine as follows:

1. The route is evaluated against the first term in the first routing policy. If the route does not match all match conditions specified before the subroutine, the subroutine is skipped and the next term in the routing policy is evaluated (see Step 2). If the route matches all match conditions specified before the subroutine, the route is evaluated against the subroutine. If the route matches the match conditions in any of the subroutine terms, two levels of evaluation occur in the following order:

   a. The actions in the subroutine term are evaluated. If one of the actions is accept, evaluation of the subroutine ends and a Boolean value of TRUE is returned to the calling policy. If one of the actions is reject, evaluation of the subroutine ends and FALSE is returned to the calling policy.

      If the subroutine does not specify the **accept**, **reject** or **next-policy** action, it uses the **accept** or **reject** action specified by the default policy, and the values of TRUE or FALSE are returned to the calling policy as described in the previous paragraph.

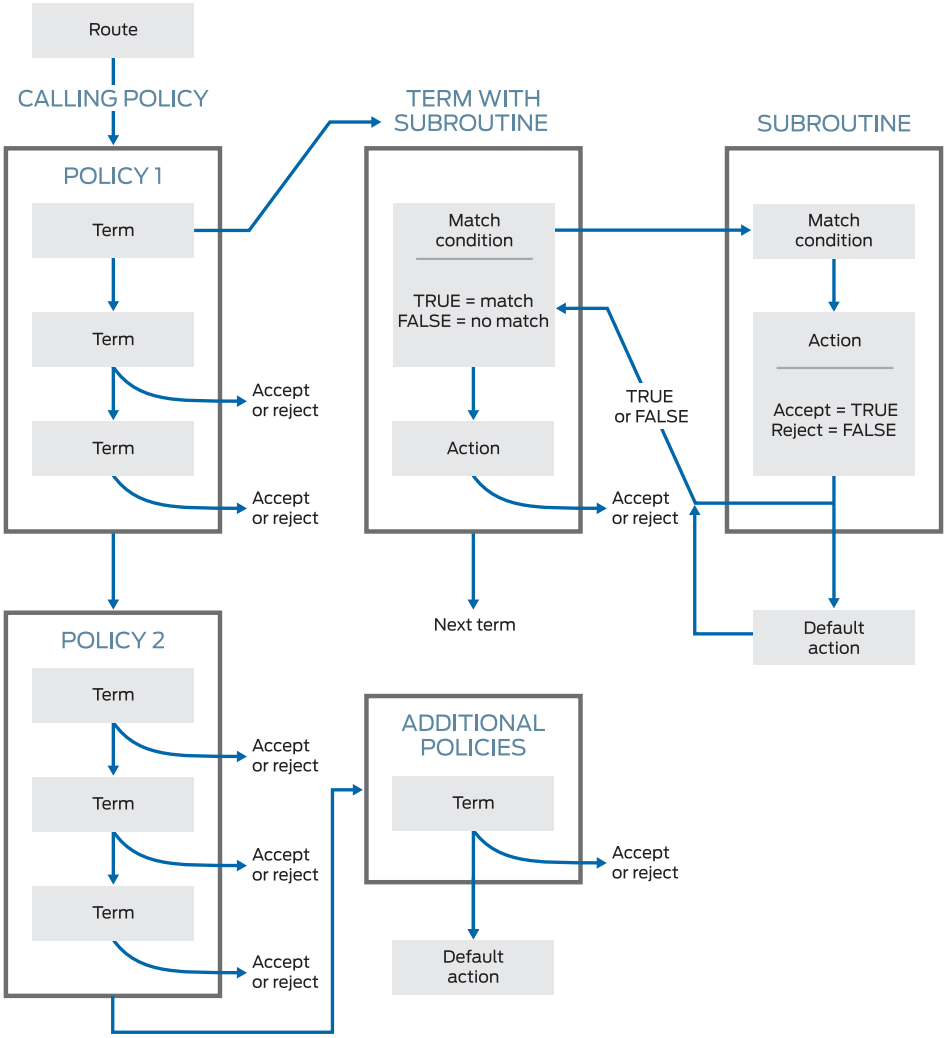
   b. The calling policy's subroutine match condition is evaluated. During this part of the evaluation, TRUE equals a match and FALSE equals no match. If the subroutine returns TRUE to the calling policy, then the evaluation of the calling policy continues. If the subroutine returns FALSE to the calling policy, then the evaluation of the current term ends and the next term is evaluated.

2. The route is evaluated against the second term in the first routing policy.

If you specify a policy chain as a subroutine, the entire chain acts as a single subroutine. As with other chains, the action specified by the default policy is taken only when the entire chain does not accept or reject a route.

If a term defines multiple match conditions, including a subroutine, and a route does not match a condition specified before the subroutine, the evaluation of the term ends and the subroutine is not called and evaluated. In this situation, an action specified in the subroutine that manipulates a route's characteristics is not implemented.

Figure 15: Routing Policy Subroutine Evaluation



Related Documentation

- Default Routing Policies on page 15
- Understanding Policy Subroutines in Routing Policy Match Conditions on page 51
- How a Routing Policy Chain Is Evaluated on page 49
- Example: Configuring a Policy Subroutine on page 273

CHAPTER 7

AS Paths

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 57
- Understanding Prepending AS Numbers to BGP AS Paths on page 63
- Understanding Adding AS Numbers to BGP AS Paths on page 64

Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions

A BGP *AS path* is a path to a destination. It is a route attribute used by BGP for both for route selection and to prevent potential routing loops. You can define regular expressions and use those expressions to locate a set of routes. An AS path consists of the AS numbers of networks that a packet traverses if it takes the associated route to a destination. The AS numbers are assembled in a sequence, or path, that is read from right to left. For example, for a packet to reach a destination using a route with an AS path 5 4 3 2 1, the packet first traverses AS 1 and so on until it reaches AS 5, which is the last AS before its destination.

You can define a match condition based on all or portions of the AS path. To do this, you create a named AS path regular expression and then include it in a routing policy.

The following sections discuss the following tasks for configuring AS path regular expressions and provides the following examples:

- Configuring AS Path Regular Expressions on page 57
- How AS Path Regular Expressions Are Evaluated on page 62
- Examples: Configuring AS Path Regular Expressions on page 62

Configuring AS Path Regular Expressions

You can create a named AS path regular expression and then include it in a routing policy with the **as-path** match condition (described in "Routing Policy Match Conditions" on page 107). To create a named AS path regular expression, include the **as-path** statement:

```
[edit policy-options]
as-path name regular-expression;
```

To include the AS path regular expression in a routing policy, include the **as-path** match condition in the **from** statement.

Additionally, you can create a named AS path group made up of AS path regular expressions and then include it in a routing policy with the **as-path-group** match condition. To create a named AS path group, include the **as-path-group** statement.

```
[edit policy-options]
  as-path-group group-name  {
  name [ regular-expressions ];
}
```

To include the AS path regular expressions within the AS path group in a routing policy, include the **as-path-group** match condition in the **from** statement.

NOTE: You cannot include both of the **as-path** and **as-path-group** statements in the same policy term.

NOTE: You can include the names of multiple AS path regular expressions in the **as-path** match condition in the **from** statement. If you do this, only one AS path regular expression needs to match for a match to occur. The AS path regular expression matching is effectively a logical OR operation.

The AS path name identifies the regular expression. It can contain letters, numbers, and hyphens (-), and can be up to 65,536 characters. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

The regular expression is used to match all or portions of the AS path. It consists of two components, which you specify in the following format:

term <operator>

- ***term***—Identifies an AS. You can specify it in one of the following ways:
  - AS number—The entire AS number composes one term. You cannot reference individual characters within an AS number, which differs from regular expressions as defined in POSIX 1003.2.
  - Wildcard character—Matches any single AS number. The wildcard character is a period (.). You can specify multiple wildcard characters.
  - AS path—A single AS number or a group of AS numbers enclosed in parentheses. Grouping the regular expression in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped path can itself include operators.

    In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. You can configure a value in the range from 1 through 4,294,967,295.
- ***operator***—(Optional) An operator specifying how the term must match. Most operators describe how many times the term must be found to be considered a match (for example, any number of occurrences, or zero, or one occurrence). Table 12 on page 59

lists the regular expression operators supported for AS paths. You place operators immediately after ***term*** with no intervening space, except for the pipe (|) and dash (–) operators, which you place between two terms, and parentheses, with which you enclose terms.

You can specify one or more term–operator pairs in a single regular expression.

Table 13 on page 59 shows examples of how to define regular expressions to match AS paths.

Table 12: AS Path Regular Expression Operators

Operator	Match Definition
{*m*,*n*}	At least ***m*** and at most ***n*** repetitions of ***term***. Both ***m*** and ***n*** must be positive integers, and ***m*** must be smaller than ***n***.
{*m*}	Exactly ***m*** repetitions of ***term***. ***m*** must be a positive integer.
{*m*,}	***m*** or more repetitions of ***term***. ***m*** must be a positive integer.
*	Zero or more repetitions of ***term***. This is equivalent to {0,}.
+	One or more repetitions of ***term***. This is equivalent to {1,}.
?	Zero or one repetition of ***term***. This is equivalent to {0,1}.
|	One of two terms on either side of the pipe.
–	Between a starting and ending range, inclusive.
^	A character at the beginning of a community attribute regular expression. This character is added implicitly; therefore, the use of it is optional.
$	A character at the end of a community attribute regular expression. This character is added implicitly; therefore, the use of it is optional.
( )	A group of terms that are enclosed in the parentheses. Intervening space between the parentheses and the terms is ignored. If a set of parentheses is enclosed in quotation marks with no intervening space "()", it indicates a null path.
[ ]	Set of AS numbers. One AS number from the set must match. To specify the start and end of a range, use a hyphen (-). A carrot (^) may be used to indicate that it does not match a particular AS number in the set, for example [^123].

Table 13: Examples of AS Path Regular Expressions

AS Path to Match	Regular Expression	Sample Matches
AS path is 1234	1234	1234

Table 13: Examples of AS Path Regular Expressions *(continued)*

AS Path to Match	Regular Expression	Sample Matches
Zero or more occurrences of AS number 1234	1234*	1234 1234 1234 1234 1234 1234 Null AS path
Zero or one occurrence of AS number 1234	1234? or 1234{0,1}	1234 Null AS path
One through four occurrences of AS number 1234	1234{1,4}	1234 1234 1234 1234 1234 1234 1234 1234 1234 1234
One through four occurrences of AS number 12, followed by one occurrence of AS number 34	12{1,4} 34	12 34 12 12 34 12 12 12 34 12 12 12 12 34
Range of AS numbers to match a single AS number	123–125	123 124 125
	[123–125]*	Null AS path 123 124 124 125 125 125 123 124 125 123
Path whose second AS number must be 56 or 78	(. 56) | (. 78) or . (56 | 78)	1234 56 1234 78 9876 56 3857 78

Table 13: Examples of AS Path Regular Expressions *(continued)*

AS Path to Match	Regular Expression	Sample Matches
Path whose second AS number might be 56 or 78	. (56 | 78) ?	1234 56 52 34 56 1234 1234 78 39 794 78 2
Path whose first AS number is 123 and second AS number is either 56 or 78	123 (56|78)	123 56 123 78
Path of any length, except nonexistent, whose second AS number can be anything, including nonexistent	. .* or . .{0,}	1234123456781234567 8
AS path is 1 2 3	1 2 3	1 2 3
One occurrence of the AS numbers 1 and 2, followed by one or more occurrences of the AS number 3	1 2 3+	1 2 3 1 2 3 3 1 2 3 3 3
One or more occurrences of AS number 1, followed by one or more occurrences of AS number 2, followed by one or more occurrences of AS number 3	1+ 2+ 3+	1 2 3 1 1 2 3 1 1 2 2 3 1 1 2 2 3 3
Path of any length that begins with AS numbers 4, 5, 6	4 5 6 .*	4 5 6 4 5 6 7 8 9
Path of any length that ends with AS numbers 4, 5, 6	.* 4 5 6	4 5 6 1 2 3 4 5 6 4 9 4 5 6
AS path 5, 12, or 18	5 | 12 | 18	5 12 18

Configuring a Null AS Path

You can use AS path regular expressions to create a null AS path that matches routes (prefixes) that have originated in your AS. These routes have not been advertised to your

AS by any external peers. To create a null AS path, use the parentheses operator enclosed in quotation marks with no intervening spaces:

```
"()"
```

In the following example, locally administered AS 2 is connected to AS 1 (10.2.2.6) and AS 3. AS 3 advertises its routes to AS 2, but the administrator for AS 2 does not want to advertise AS 3 routes to AS 1 and thereby allow transit traffic from AS 1 to AS 3 through AS 2. To prevent transit traffic, the export policy **only-my-routes** is applied to AS 1. It permits advertisement of routes from AS 2 to AS 1 but prevents advertisement of routes for AS 3 (or routes for any other connected AS) to AS 1:

```
[edit policy-options]
null-as "()";
policy-statement only-my-routes {
   term just-my-as {
      from {
         protocol bgp;
         as-path null-as;
      }
      then accept;
   }
   term nothing-else {
      then reject;
   }
}
protocol {
   bgp {
      neighbor 10.2.2.6 {
         export only-my-routes;
      }
   }
}
```

How AS Path Regular Expressions Are Evaluated

AS path regular expressions implement the extended (modern) regular expressions as defined in POSIX 1003.2. They are identical to the UNIX regular expressions with the following exceptions:

- The basic unit of matching in an AS path regular expression is the AS number and not an individual character.
- A regular expression matches a route only if the AS path in the route exactly matches ***regular-expression***. The equivalent UNIX regular expression is **^*regular-expression*$**. For example, the AS path regular expression **1234** is equivalent to the UNIX regular expression **^1234$**.
- You can specify a regular expression using wildcard operators.

Examples: Configuring AS Path Regular Expressions

Exactly match routes with the AS path 1234 56 78 9 and accept them:

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

```
    as-path wellington "1234 56 78 9";
    policy-statement from-wellington {
      term term1 {
        from as-path wellington;
      }
        then {
          preference 200;
          accept;
        }
      term term2 {
        then reject;
      }
    }
}
```

Match alternate paths to an AS and accept them after modifying the preference:

```
[edit]
policy-options {
  as-path wellington-alternate “1234{1,6} (56|47)? (78|101|112)* 9+”;
  policy-statement from-wellington {
    from as-path wellington-alternate;
    }
    then {
      preference 200;
      accept;
    }
  }
}
```

Match routes with an AS path of 123, 124, or 125 and accept them after modifying the preference:

```
[edit]
policy-options {
  as-path addison "123–125";
  policy-statement from-addison {
    from as-path addison;
    }
    then {
      preference 200;
      accept;
    }
  }
}
```

Related Documentation

- Example: Using AS Path Regular Expressions on page 283
- Example: Configuring a Routing Policy to Prepend the AS Path on page 292

Understanding Prepending AS Numbers to BGP AS Paths

You can *prepend* one or more autonomous system (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.

The BGP best path algorithm determines how the best path to an autonomous system (AS) is selected. The AS path length determines the best path when all of the following conditions are met:

- There are multiple potential routes to an AS.
- BGP has the lowest preference value (sometimes referred to as administrative distance) of the available routes.
- The local preferences of the available routes are equal.

When these conditions are met, the AS path length is used as the tie breaker in the best path algorithm. When two or more routes exist to reach a particular prefix, BGP prefers the route with the shortest AS Path length.

If you are an enterprise that has multihoming to one or more service providers, you might prefer that incoming traffic take a particular path to reach your network. Perhaps you have two connections, but one costs less than the other. Or you might have one fast connection and another, much slower connection that you only want to use as a backup if your primary connection is down. AS path prepending is an easy method that you can use to influence inbound routing to your AS.

In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295.

If you have a router that does not support 4-byte AS numbers in the AS path, the prependend AS number displayed in the AS path is the AS_TRANS number, AS 23456. To display the route details, use the *show route* command.

Related Documentation

- Example: Configuring a Routing Policy to Prepend the AS Path on page 292
- Example: Using AS Path Regular Expressions on page 283
- *Understanding BGP Path Selection*

Understanding Adding AS Numbers to BGP AS Paths

You can expand or add one or more AS numbers to an AS sequence. The AS numbers are added before the local AS number has been added to the path. Expanding an AS path makes a shorter AS path look longer and therefore less preferable to BGP. The last AS number in the existing path is extracted and prepended *n* times, where *n* is a number from 1 through 32. This is similar to the AS path prepend action, except that the AS path expand action adds an arbitrary sequence of AS numbers.

For example, from AS 1 there are two equal paths (through AS 2 and AS 3) to reach AS 4. You might want packets from certain sources to use the path through AS 2. Therefore,

you must make the path through AS 3 less preferable so that BGP chooses the path through AS 2. In AS 1, you can expand multiple AS numbers.

```
[edit]
policy-options {
   policy-statement as-path-expand {
      term expand {
         from {
            route-filter 192.168.0.0/16 orlonger;
            route-filter 172.16.0.0/12 orlonger;
            route-filter 10.0.0.0/8 orlonger;
         }
         then as-path-expand last-as count 4;
      }
   }
}
```

For routes from AS 2, this makes the route look like 1 2 2 2 2 2 when advertised, where 1 is from AS 1, the 2 from AS 2 is prepended four times, and the final 2 is the original 2 received from the neighbor router.

Related Documentation

- Example: Advertising Multiple Paths in BGP on page 301
- Example: Configuring a Routing Policy to Prepend the AS Path on page 292

CHAPTER 8

Communities

- Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions on page 67
- Understanding How to Define BGP Communities and Extended Communities on page 68
- How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions on page 74

Understanding BGP Communities and Extended 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.

Related Documentation

- Understanding How to Define BGP Communities and Extended Communities on page 68
- How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions on page 74
- Example: Defining a Routing Policy That Removes BGP Communities on page 358
- Example: Configuring Communities in a Routing Policy on page 327
- Example: Configuring Extended Communities in a Routing Policy on page 342
- *Example: Redistributing BGP Routes with a Specific Community Tag into IS-IS*

Understanding How to Define BGP Communities and Extended Communities

To use a BGP community or extended community as a routing policy match condition, you define the community as described in the following sections:

- Defining BGP Communities for Use in Routing Policy Match Conditions on page 68
- Defining BGP Extended Communities for Use in Routing Policy Match Conditions on page 72

Defining BGP Communities for Use in Routing Policy Match Conditions

To create a named BGP community and define the community members, include the **community** statement:

```
[edit policy-options]
community name {
    invert-match;
    members [ community-ids ];
}
```

name identifies the community. It 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 quotation marks (“ ”).

community-ids identifies one or more members of the community. Each community ID consists of two components, which you specify in the following format:

as-number:*community-value*;

- *as-number*—AS number of the community member. It can be a value from 0 through 65,535. You can use the following notation in specifying the AS number:
  - String of digits.
  - Asterisk (*)—A wildcard character that matches all AS numbers. (In the definition of the community attribute, the asterisk also functions as described in Table 14 on page 70.)
  - Period (.)—A wildcard character that matches any single digit in an AS number.
  - Group of AS numbers—A single AS number or a group of AS numbers enclosed in parentheses. Grouping the numbers in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped numbers can themselves include regular expression operators. For more information about regular expressions, see “Using UNIX Regular Expressions in Community Names” on page 70.
- *community-value*—Identifier of the community member. It can be a number from 0 through 65,535. You can use the following notation in specifying the community ID:
  - String of digits.
  - Asterisk (*)—A wildcard character that matches all community values. (In the definition of the community attribute, the asterisk also functions as described in Table 14 on page 70.)
  - Period (.)—A wildcard character that matches any single digit in a community value number.
  - Group of community value numbers—A single community value number or a group of community value numbers enclosed in parentheses. Grouping the regular expression in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped path can itself include regular expression operators.

You can also include one of the following well-known community names (defined in RFC 1997, *BGP Communities Attribute*) in the *community-ids* option for the **members** statement:

- no-advertise—Routes in this community name must not be advertised to other BGP peers.
- no-export—Routes in this community must not be advertised outside a BGP confederation boundary.

- no-export-subconfed—Routes in this community must not be advertised to external BGP peers, including peers in other members' ASs inside a BGP confederation.

Using UNIX Regular Expressions in Community Names

When specifying the members of a named BGP community (in the **members [** ***community-ids*** **]** statement), you can use UNIX-style regular expressions to specify the AS number and the member identifier. A regular expression consists of two components, which you specify in the following format:

term operator;

term identifies the string to match.

operator specifies how the term must match. Table 14 on page 70 lists the regular expression operators supported in community IDs. You place an operator immediately after *term* with no intervening space, except for the pipe (**|**) and dash (**–**) operators, which you place between two terms, and parentheses, with which you enclose terms. Table 15 on page 71 shows examples of how to define ***community-ids*** using community regular expressions. The operator is optional.

Community regular expressions are identical to the UNIX regular expressions. Both implement the extended (or modern) regular expressions as defined in POSIX 1003.2.

Community regular expressions evaluate the string specified in ***term*** on a character-by-character basis. For example, if you specify **1234:5678** as ***term***, the regular expressions see nine discrete characters, including the colon (**:**), instead of two sets of numbers (**1234** and **5678**) separated by a colon.

NOTE: In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS.

Table 14: Community Attribute Regular Expression Operators

Operator	Match Definition
{*m*,*n*}	At least ***m*** and at most ***n*** repetitions of ***term***. Both ***m*** and ***n*** must be positive integers, and ***m*** must be smaller than ***n***.
{*m*}	Exactly ***m*** repetitions of ***term***. ***m*** must be a positive integer.
{*m*,}	***m*** or more repetitions of ***term***. ***m*** must be a positive integer.
*	Zero or more repetitions of ***term***. This is equivalent to **{0,}**.
+	One or more repetitions of ***term***. This is equivalent to **{1,}**.
?	Zero or one repetition of ***term***. This is equivalent to **{0,1}**.

Table 14: Community Attribute Regular Expression Operators *(continued)*

Operator	Match Definition
|	One of the two terms on either side of the pipe.
–	Between a starting and ending range, inclusive.
^	Character at the beginning of a community attribute regular expression. If you omit the ^ character, it is implicitly added. We recommend explicit use of this operator for the clearest interpretation of your configuration.
$	Character at the end of a community attribute regular expression. If you omit the $ character, it is implicitly added. We recommend explicit use of this operator for the clearest interpretation of your configuration.
[ ]	Set of characters. One character from the set can match. To specify the start and end of a range, use a hyphen (**-**). To specify a set of characters that do not match, use the caret (**^**) as the first character after the opening square bracket (**[**).
( )	Group of terms that are enclosed in parentheses. If enclosed in quotation marks with no intervening space ("**()**"), indicates a null. Intervening space between the parentheses and the terms is ignored.
“ ”	Characters (such as space, tab, question mark, and bracket) that are enclosed within quotation marks in a community attribute regular expression indicate special characters.

Table 15: Examples of Community Attribute Regular Expressions

Community Attribute to Match	Regular Expression	Sample Matches
AS number is 56 or 78. Community value is any number.	^((56) | (78)):(.*)$	56:1000 78:65000
AS number is 56. Community value is any number that starts with 2.	^56:(2.*)$	56:2 56:222 56:234
AS number is any number. Community value is any number that ends with 5, 7, or 9.	^(.*):(.*[579])$	1234:5 78:2357 34:65009

Table 15: Examples of Community Attribute Regular Expressions *(continued)*

Community Attribute to Match	Regular Expression	Sample Matches
AS number is 56 or 78. Community value is any number that starts with 2 and ends with 2 through 8.	^((56) | (78)):(2.*[2–8])$	56:22 56:21197 78:2678

Defining BGP Extended Communities for Use in Routing Policy Match Conditions

To create a named BGP community and define the community members, include the **community** statement:

```
[edit policy-options]
community name {
  members [ community-ids ];
}
```

name identifies the community. It 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 quotation marks (“ ”).

community-ids identifies one or more members of the community. Each community ID consists of three components, which you specify in the following format:

type:*administrator*:*assigned-number*

type is the type of extended community and can be either the 16-bit numerical identifier of a specific BGP extended community or one of these types:

- **bandwidth**—Sets up the bandwidth extended community. Specifying link bandwidth allows you to distribute traffic unequally among different BGP paths.

  NOTE: The link bandwidth attribute does not work concurrently with per-prefix load balancing.

- **domain-id**—Identifies the OSPF domain from which the route originated.
- **origin**—Identifies where the route originated.
- **rt-import**—Identifies the route to install in the routing table.

  NOTE: You must identify the route by an IP address, not an AS number.

- **src-as**—Identifies the AS from which the route originated. You must specify an AS number, not an IP address.

NOTE: You must identify the AS by an AS number, not an IP address.

- **target**—Identifies the destination to which the route is going.

NOTE: For an import policy for a VPN routing and forwarding (VRF) instance, you must include at least one route target. Additionally, you cannot use wildcard characters or regular expressions in the route target for a VRF import policy. Each value you configure for a route target for a VRF import policy must be a single value.

administrator is the administrator. It is either an AS number or an IP version 4 (IPv4) address prefix, depending on the type of extended community.

assigned-number identifies the local provider.

In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295. To configure a **target** or **origin** extended community that includes a 4-byte AS number in the plain-number format, append the letter "L" to the end of number. For example, a target community with the 4-byte AS number 334,324 and an assigned number of 132 is represented as **target:334324L:132**.

In Junos OS Release 9.2 and later, you can also use AS-dot notation when defining a 4-byte AS number for the **target** and **origin** extended communities. Specify two integers 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 represented in plain-number format as 65546 is represented in AS-dot notation as 1.10.

Examples: Defining BGP Extended Communities

Configure a target community with an administrative field of **10458** and an assigned number of **20**:

```
[edit policy-options]
community test-a members [ target:10458:20 ];
```

Configure a target community with an administrative field of 10.1.1.1 and an assigned number of 20:

```
[edit policy-options]
community test-a members [ target:10.1.1.1:20 ];
```

Configure an origin community with an administrative field of 10.1.1.1 and an assigned number of 20:

```
[edit policy-options]
community test-a members [ origin:10.1.1.1:20 ];
```

Configure a target community with a 4-byte AS number in the administrative field of 100000 and an assigned number of 130:

```
[edit policy-options]
community test-b members [ target:100000L:130 ];
```

Related Documentation

- Example: Configuring Communities in a Routing Policy on page 327
- Example: Configuring Extended Communities in a Routing Policy on page 342

How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions

When you use BGP communities and extended communities as match conditions in a routing policy, the policy framework software evaluates them as follows:

- Each route is evaluated against each named community in a routing policy **from** statement. If a route matches one of the named communities in the **from** statement, the evaluation of the current term continues. If a route does not match, the evaluation of the current term ends.
- The route is evaluated against each member of a named community. The evaluation of all members must be successful for the named community evaluation to be successful.
- Each member in a named community is identified by either a literal community value or a regular expression. Each member is evaluated against each community associated with the route. (Communities are an unordered property of a route. For example, 1:2 3:4 is the same as 3:4 1:2.) Only one community from the route is required to match for the member evaluation to be successful.
- Community regular expressions are evaluated on a character-by-character basis. For example, if a route contains community 1234:5678, the regular expressions see nine discrete characters, including the colon (:), instead of two sets of numbers (1234 and 5678) separated by a colon. For example:

  ```
  [edit]
  policy-options {
     policy-statement one {
        from {
           community [comm-one comm-two];
        }
     }
     community comm-one members [ 1:2 "^4:(5|6)$" ];
     community comm-two members [ 7:8 9:10 ];
  }
  ```

  If a community member is a regular expression, a string match is made rather than a numeric match.

  For example:

  ```
  community example1 members 100:100
  community example2 members 100:1..
  ```

Given a route with a community value of 1100:100, this route matches **community example2** but not **example1**.

- To match routing policy **one**, the route must match either **comm-one** or **comm-two**.
- To match **comm-one**, the route must have a community that matches 1:2 and a community that matches 4:5 or 4:6.
- To match **comm-two**, the route must have a community that matches 7:8 and a community that matches 9:10.

Multiple Matches

When multiple matches are found, label aggregation does not happen. Consider the following configuration:

```
family inet-vpn {
  unicast {
    aggregate-label {
      community community-name;
    }
  }
}
family inet-vpn {
  labeled-unicast {
    aggregate-label {
      community community-name;
    }
  }
}
```

Suppose, for instance, that two routes are received with community attributes **target:65000:1000 origin:65200:2000** and that the community name is **"5...:.*"**. In this case, both the extended community attributes, **target:65000:1000** and **origin:65200:2000** match the regular expression of the community name. In this case, label aggregation does not occur. In the following example, the **Label operation** field shows that the labels are not aggregated.

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101056
                Push 101056
                Communities: target:65000:1000 origin:65200:2000
```

You can resolve this issue in either of the following ways:

- Be more specific in the regular expression if the site-of-origin extended community attribute does not overlap with the target one.
- Specify the site of origin in the community name.

Both methods are shown in the following examples.

Be More Specific in the Regular Expression

```
user@host# set policy-options community community-name members "52...:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
```

Specify the Site of Origin in the Community Name

```
user@host# set policy-options community community-name members "origin:65...:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
```

Inverting Community Matches

The **community** match condition defines a regular expression and if it matches the community attribute of the received prefix, Junos OS returns a TRUE result. If not, Junos OS returns a FALSE result. The **invert-match** statement makes Junos OS behave to the contrary. If there is a match, Junos OS returns a FALSE result. If there is no match, Junos OS returns a TRUE result. To invert the results of the community expression matching, include the **invert-match** statement in the community configuration.

```
[edit policy-options community name]
invert-match;
```

Extended Community Type

The extended community type is not taken into account by regular expressions. Consider, for instance, the following community attributes and community name.

Communities:

- 5200:1000
- **target:65000:1000**
- **origin:65200:2000**

Community attribute:

- community-name members "5...:.*"

In this case, both extended community attribute, **5200:1000** and the extended community attribute, **origin:65200:2000**, match the regular expression of the community name. Therefore, the label aggregation does not occur, as shown here:

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101056
                Push 101056
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
```

You can resolve this issue by using a more specific regular expression. For example, you can use the anchor character (^) to bind the location of the digits, as shown here:

```
user@host# set policy-options community community-name members "^5...:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
```

Multiple Communities Are Matched with Ex-OR Logic

This differs from the AND matching logic used for non-extended communities in BGP.

If, for instance, four routes are received with two sets of community attributes, the regular expression might match both community attributes. Consider the following example:

- Communities—5200:1000 target:65000:1000
- Communities—target:65000:1000 origin:65200:2000
- Community attribute—community community-name member ["^5...:.*" origin:65.*:.*]

Both labels are aggregated, as shown here:

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
```

```
10.1.1.16/30 (1 entry, 1 announced)
                Label operation: Push 121104
                Push 101104
                Communities: 5200:1000 target:65000:1000
10.1.1.20/30 (1 entry, 1 announced)
                Label operation: Push 121104
                Push 101104
                Communities: 5200:1000 target:65000:1000
```

A more complete example of community values is shown here:

```
user@host> show policy-options community community-name
members [ "(^1...:*)|(^3...:*)|(^4...:*)" origin:2.*:* origin:3.*:* origin:6.*:*
 ]
```

This regular expression matches community values starting with 1, 3, or 4, and matches extended community values of type origin whose administrative value starts with 2, 3, or 6.

Including BGP Communities and Extended Communities in Routing Policy Match Conditions

To include a BGP community or extended community in a routing policy match condition, include the **community** condition in the **from** statement of a policy term:

```
from {
  community [ names ];
}
```

Additionally, you can explicitly exclude BGP community information with a static route by using the **none** option. Include this option when configuring an individual route in the **route** portion to override a community option specified in the **defaults** portion.

You can include the names of multiple communities in the **community** match condition. If you do this, only one community needs to match for a match to occur (matching is effectively a logical OR operation).

Related Documentation

- Using UNIX Regular Expressions in Community Names on page 70
- Example: Configuring Communities in a Routing Policy on page 327
- Example: Configuring Extended Communities in a Routing Policy on page 342
- Example: Defining a Routing Policy That Removes BGP Communities on page 358
- Example: Defining a Routing Policy Based on the Number of BGP Communities on page 351

CHAPTER 9

Testing Policies

- Understanding Routing Policy Tests on page 79

Understanding Routing Policy Tests

Routing policy tests provide a method for verifying the effectiveness of your policies before applying them on the routing device. Before applying a routing policy, you can issue the **test policy** command to ensure that the policy produces the results that you expect:

```
user@host> test policy policy-name prefix
```

Keep in mind that different protocols have different default policies that get applied if the prefix does not match the configured policy. For BGP this is accept, but for RIP it is reject. The **test policy** command always uses accept as the default policy, so unless you explicitly reject all routes that you do not want to match you might see more routes matching than you want.

The default policy of the **test policy** command accepts all routes from all protocols. Test output can be misleading when you are evaluating protocol-specific conditions. For example, if you define a policy for BGP that accepts routes of a specified prefix and apply it to BGP as an export policy, BGP routes that match the prefix are advertised to BGP peers. However, if you test the same policy using the **test policy** command, the test output might indicate that non-BGP routes have been accepted.

Example: Testing a Routing Policy

Test the following policy, which looks for unwanted routes and rejects them:

```
[edit policy-options]
policy-statement reject-unwanted-routes {
   term drop-these-routes {
      from {
         route-filter 0/0 exact;
         route-filter 10/8 orlonger;
         route-filter 172.16/12 orlonger;
         route-filter 192.168/16 orlonger;
         route-filter 224/3 orlonger;
      }
      then reject;
   }
```

```
}
```

Test this policy against all routes in the routing table:

```
user@host> test policy reject-unwanted-routes 0/0
```

Test this policy against a specific set of routes:

```
user@host> test policy reject-unwanted-routes 10.49.0.0/16
```

Related Documentation

- Example: Testing a Routing Policy with Complex Regular Expressions on page 367

CHAPTER 10

Damp BGP Route Flapping

- Understanding Damping Parameters on page 81
- Using Routing Policies to Damp BGP Route Flapping on page 82

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.) The exception to this rule is when flap damping is applied at the address family level, which is supported in Junos OS Release 12.2 and later. 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 16 on page 81.

Table 16: 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 4
max-suppress ***minutes***	Maximum hold-down time for a route, in minutes.	60 (minutes)	1 through 720

Table 16: Damping Parameters *(continued)*

Damping Parameter	Description	Default Value	Possible Values
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.

Related Documentation

- *Junos OS Feature Support Reference for SRX Series and J Series Devices*
- Understanding Routing Policies on page 12
- Example: Configuring Damping Parameters on page 375
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 384

Using Routing Policies to Damp BGP Route Flapping

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 way to reduce the number of update messages sent between BGP peers, thereby reducing the load on these peers without adversely affecting the route convergence time.

Flap damping reduces the number of update messages by marking routes as ineligible for selection as the active or preferable route. Doing this 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 (that is, to routes in different ASs). You can also apply it within a confederation, between confederation member ASs. Because routing consistency within an AS is important, do not apply flap damping to IBGP routes. (If you do, it is ignored.)

BGP flap damping is defined in RFC 2439, *BGP Route Flap Damping.*

To effect changes to 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 (described in "Configuring Actions That Manipulate Route Characteristics" on page 121). For the damping routing policy to work, you also must enable BGP route flap damping.

The following sections discuss the following topics:

- Configuring BGP Flap Damping Parameters on page 83
- Specifying BGP Flap Damping as the Action in Routing Policy Terms on page 85

- Disabling Damping for Specific Address Prefixes on page 85
- Example: Configuring BGP Flap Damping on page 86

Configuring BGP Flap Damping Parameters

To define damping parameters, include the **damping** statement:

```
[edit policy-options]
damping name {
    disable;
    half-life minutes;
    max-suppress minutes;
    reuse number;
    suppress number;
}
```

The name identifies the group of damping parameters. It can contain letters, numbers, and hyphens (-) and can be up to 255 characters. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

You can specify one or more of the damping parameters described in Table 17 on page 83.

Table 17: Damping Parameters

Damping Parameter	Description	Default	Possible Values
half-life *minutes*	Decay half-life, in minutes	15 minutes	1 through 45 minutes
max-suppress *minutes*	Maximum hold-down time, in minutes	60 minutes	1 through 720 minutes
reuse	Reuse threshold	750 (unitless)	1 through 20,000 (unitless)
suppress	Cutoff (suppression) threshold	3000 (unitless)	1 through 20,000 (unitless)

If you do not specify one or more of the damping parameters, the default value of the parameter is used.

To understand how to configure these parameters, you need to understand how damping suppresses routes. How long a route can be suppressed is based on a *figure of merit*, which is a value that correlates to the probability of future instability of a route. Routes with higher figure-of-merit values are suppressed for longer periods of time. The figure-of-merit value decays exponentially over time.

A figure-of-merit value of zero is assigned to each new route. The value is increased each time the route is withdrawn or readvertised, or when one of its path attributes changes. With each incident of instability, the value increases as follows:

- Route is withdrawn—1000
- Route is readvertised—1000

- Route's path attributes change—500

NOTE: Other vendors' implementations for figure-of-merit increase the value only when a route is withdrawn. The Junos OS implementation for figure-of-merit increases the value for both route withdrawal and route readvertisement. To accommodate other implementations for figure-of-merit, multiply the reuse and suppress threshold values by 2.

When a route's figure-of-merit value reaches a particular level, called the *cutoff* or *suppression threshold*, the route is suppressed. If a route is suppressed, the routing table no longer installs the route into the forwarding table and no longer exports this route to any of the routing protocols. By default, a route is suppressed when its figure-of-merit value reaches 3000. To modify this default, include the **suppress** option at the **[edit policy-options damping *name*]** hierarchy level.

If a route has flapped, but then becomes stable so that none of the incidents listed previously occur within a configurable amount of time, the figure-of-merit value for the route decays exponentially. The default half-life is 15 minutes. For example, for a route with a figure-of-merit value of 1500, if no incidents occur, its figure-of-merit value is reduced to 750 after 15 minutes and to 375 after another 15 minutes. To modify the default half-life, include the **half-life** option at the **[edit policy-options damping *name*]** hierarchy level.

NOTE: For the half-life, configure a value that is less than the max-suppress. If you do not, the configuration is rejected.

A suppressed route becomes reusable when its figure-of-merit value decays to a value below a *reuse threshold*, thus allowing routes that experience transient instability to once again be considered valid. The default reuse threshold is 750. When the figure-of-merit value passes below the reuse threshold, the route once again is considered usable and can be installed in the forwarding table and exported from the routing table. To modify the default reuse threshold, include the **reuse** option at the **[edit policy-options damping *name*]** hierarchy level.

The maximum suppression time provides an upper bound on the time that a route can remain suppressed. The default maximum suppression time is 60 minutes. To modify the default, include the **max-suppress** option at the **[edit policy-options damping *name*]** hierarchy level.

NOTE: For the max-suppress, configure a value that is greater than the half-life. If you do not, the configuration is rejected.

A route's figure-of-merit value stops increasing when it reaches a maximum suppression threshold, which is determined based on the route's suppression threshold level, half-life, reuse threshold, and maximum hold-down time.

The merit ceiling, ε_c, which is the maximum merit that a flapping route can collect, is calculated using the following formula:

$$\varepsilon_c \leq \varepsilon_r e^{(t/\lambda)(\ln 2)}$$

ε_r is the figure-of-merit reuse threshold, t is the maximum hold-down time in minutes, and λ is the half-life in minutes. For example, if you use the default figure-of-merit values in this formula, but use a half-life of 30 minutes, the calculation is as follows:

$$\varepsilon_c \leq 750\, e^{(60/30)(\ln 2)}$$

$$\varepsilon_c \leq 3000$$

NOTE: The cutoff threshold, which you configure using the **suppress** option, must be less than or equal to the merit ceiling, ε_c. If the configured cutoff threshold or the default cutoff threshold is greater than the merit ceiling, the route is never suppressed and damping never occurs.

To display figure-of-merit information, use the **show policy damping** command.

A route that has been assigned a figure of merit is considered to have a damping state. To display the current damping information on the routing device, use the **show route detail** command.

Specifying BGP Flap Damping as the Action in Routing Policy Terms

To BGP flap damping as the action in a routing policy term, include the **damping** statement and the name of the configured damping parameters either as an option of the **route-filter** statement at the **[edit policy-options policy-statement** *policy-name* **term** *term-name* **from]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name from]
route-filter destination-prefix match-type {
   damping damping-parameters;
}
```

or at the **[edit policy-options policy-statement** *policy-name* **term** *term-name* **then]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name then]
damping damping-parameters;
```

Disabling Damping for Specific Address Prefixes

Normally, you enable or disable damping on a per-peer basis. However, you can disable damping for a specific prefix received from a peer by including the **disable** option:

```
[edit policy-options damping name]
disable;
```

Example: Disabling Damping for a Specific Address Prefix

In this routing policy example, although damping is enabled for the peer, the **damping none** statement specifies that damping be disabled for prefix 10.0.0.0/8 in **Policy-A**. This route is not damped because the routing policy statement named **Policy-A** filters on the prefix 10.0.0.0/8 and the action points to the **damping** statement named **none**. The remaining prefixes are damped using the default parameters.

```
[edit]
policy-options {
   policy-statement Policy-A {
      from {
         route-filter 10.0.0.0/8 exact;
      }
      then damping none;
   }
   damping none {
      disable;
   }
}
```

Example: Configuring BGP Flap Damping

Enable BGP flap damping and configure damping parameters:

```
[edit]
routing-options {
   autonomous-system 666;
}
protocols {
   bgp {
      damping;
      group group1 {
         traceoptions {
            file bgp-log size 1m files 10;
            flag damping;
         }
         import damp;
         type external;
         peer-as 10458;
         neighbor 192.168.2.30;
      }
   }
}
policy-options {
   policy-statement damp {
      from {
         route-filter 192.168.0.0/32 exact {
            damping high;
            accept;
         }
         route-filter 172.16.0.0/32 exact {
            damping medium;
            accept;
         }
```

```
        route-filter 10.0.0.0/8 exact {
          damping none;
          accept;
        }
      }
    }
    damping high {
      half-life 30;
      suppress 3000;
      reuse 750;
      max-suppress 60;
    }
    damping medium {
      half-life 15;
      suppress 3000;
      reuse 750;
      max-suppress 45;
    }
    damping none {
      disable;
    }
  }
```

To display damping parameters for this configuration, use the **show policy damping** command:

```
user@host> show policy damping
Damping information for "high":
  Halflife: 30 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 60 minutes
  Computed values:
    Merit ceiling: 3008
    Maximum decay: 24933
Damping information for "medium":
  Halflife: 15 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 45 minutes
  Computed values:
    Merit ceiling: 6024
    Maximum decay: 12449
Damping information for "none":
Damping disabled
```

Related Documentation

- Example: Configuring Damping Parameters on page 375
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 384

CHAPTER 11

Source Class Usage and Destination Class Usage

- Understanding Source Class Usage and Destination Class Usage Options on page 89

Understanding Source Class Usage and Destination Class Usage Options

You can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into disjoint sets defined as source classes and *destination classes*. You can define classes based on a variety of parameters, such as routing neighbors, autonomous systems, and route filters.

Source class usage (SCU) counts packets sent to customers by performing lookups on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge. You must enable SCU accounting on both the inbound and outbound physical interfaces.

Destination class usage (DCU) counts packets from customers by performing lookups of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

On T Series Core Routers and M320 Multiservice Edge Routers, the source class and destination classes are not carried across the platform fabric. The implications of this are as follows:

- On T Series and M320 routers, SCU and DCU accounting is performed before the packet enters the fabric.
- On T Series and M320 routers, DCU is performed before output filters are evaluated. On M Series platforms, DCU is performed after output filters are evaluated.
- If an output filter drops traffic on M Series devices, the dropped packets are excluded from DCU statistics. If an output filter drops traffic on T Series and M320 routers, the dropped packets are included in DCU statistics.

On Enhanced Scaling FPCs (T640-FPC1-ES, T640-FPC2-ES, T640-FPC3-ES, T640-FPC4-1P-ES , and T1600-FPC4-ES), the source class accounting is performed at

ingress. On a T4000 Type 5 FPC, the source class accounting is performed at egress. The implications of this are as follows:

- SCU accounting is *not* performed when packets traverse from T4000 Type 5 FPC (ingress FPC) to Enhanced Scaling FPCs (egress FPC).
- SCU accounting is performed when packets traverse from Enhanced Scaling FPCs (ingress FPC) to T4000 Type 5 FPC (egress FPC).

Class-based filter match conditions are not supported on J Series Services Routers.

For more information about source class usage, see the *Routing Policy Feature Guide for Routing Devices*, the *Junos OS Network Interfaces Library for Routing Devices*, and the *Junos OS Feature Guides*.

Related Documentation

- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 395
- *Configuring SCU or DCU*
- *Configuring SCU on a Virtual Loopback Tunnel Interface*
- *Configuring Class Usage Profiles*
- *Configuring the MIB Profile*
- *Configuring the Routing Engine Profile*

CHAPTER 12

Conditional Routing Policies

- Understanding Conditional Installation of Prefixes in a Routing Table on page 92
- Conditional Installation of Prefixes Use Cases on page 94

Understanding Conditional Installation of Prefixes in a Routing Table

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:

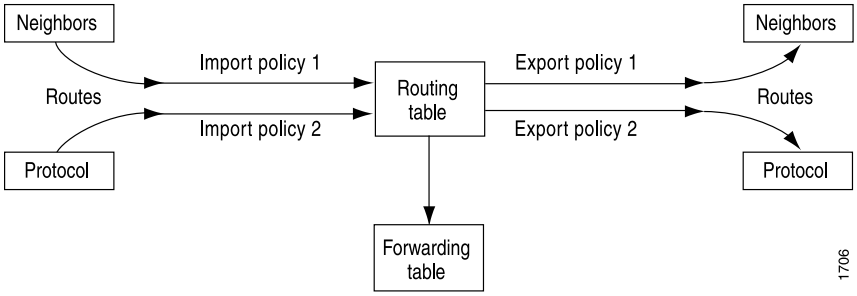
```
[edit]
policy-options {
  condition condition-name {
    if-route-exists address table table-name;
  }
}
```

This is known as conditional installation of prefixes and is described in "Example: Configuring Conditional Installation of Prefixes in a Routing Table" on page 405.

The Juniper Networks® Junos® Operating System (Junos OS) supports conditional export of routes based on the existence of another route in the routing table. Junos OS does not, however, support policy conditions for import policy.

Figure 16 on page 92 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 16: 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;
  }
}
```

Related Documentation

- Conditional Installation of Prefixes Use Cases on page 94
- Example: Configuring Conditional Installation of Prefixes in a Routing Table on page 405

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 Conditional Installation of Prefixes in a Routing Table" on page 405 explains how the conditional installation of prefixes can be configured and verified.

Related Documentation

- Understanding Conditional Installation of Prefixes in a Routing Table on page 92
- Example: Configuring Conditional Installation of Prefixes in a Routing Table on page 405

CHAPTER 13

Dynamic Routing Policies

- Understanding Dynamic Routing Policies on page 97

Understanding Dynamic Routing Policies

The verification process required to commit configuration changes can entail a significant amount of overhead and time. For example, changing a prefix in one line of a routing policy that is 20,000 lines long can take up to 20 seconds to commit. It can be useful to be able to commit routing policy changes much more quickly.

In Junos OS Release 9.5 and later, you can configure routing policies and certain routing policy objects in a dynamic database that is not subject to the same verification required in the standard configuration database. As a result, the time it takes to commit changes to the dynamic database is much shorter than for the standard configuration database. You can then reference these policies and policy objects in routing policies you configure in the standard database. BGP is the only protocol to which you can apply routing policies that reference policies and policy objects configured in the dynamic database. After you configure and commit a routing policy based on the objects configured in the dynamic database, you can quickly update any existing routing policy by making changes to the dynamic database configuration.

CAUTION: **Because the Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.**

- Configuring Routing Policies and Policy Objects in the Dynamic Database on page 97
- Configuring Routing Policies Based on Dynamic Database Configuration on page 98
- Applying Dynamic Routing Policies to BGP on page 100
- Preventing Reestablishment of BGP Peering Sessions After NSR Routing Engine Switchover on page 100

Configuring Routing Policies and Policy Objects in the Dynamic Database

Junos OS Release 9.5 and later support a configuration database, the *dynamic database*, which can be edited in a similar way to the standard configuration database but which is not subject to the same verification process to commit configuration changes. As a

result, the time it takes to commit a configuration change is much faster. The policies and policy objects defined in the dynamic database can then be referenced in routing policies configured in the standard configuration. The dynamic database is stored in the **/var/run/db/juniper.dyn** directory.

To configure the dynamic database, enter the **configure dynamic** command to enter the configuration mode for the dynamic database:

```
user@host> configure dynamic
Entering configuration mode

[edit dynamic]
user@host#
```

In this dynamic configuration database, you can configure the following statements at the **[edit policy-options]** hierarchy level:

- as-path *name*
- as-path-group *group-name*
- community *community-name*
- condition *condition-name*
- prefix-list *prefix-list-name*
- policy-statement *policy-statement-name*

NOTE: No other configuration is supported at the **[edit dynamic]** hierarchy level.

Use the **policy-statement** *policy-statement-name* statement to configure routing policies as you would in the standard configuration database.

To exit configuration mode for the dynamic database, issue the **exit configuration-mode** command from any level within the **[edit dynamic]** hierarchy, or use the **exit** command from the top level.

Configuring Routing Policies Based on Dynamic Database Configuration

In the standard configuration mode, you can configure routing policies that reference policies and policy objects configured at the **[edit dynamic]** hierarchy level in the dynamic database. To define a routing policy that references the dynamic database configuration, include the **dynamic-db** statement at the **[edit policy-options policy-statement** ***policy-statement-name*****]** hierarchy level:

```
[edit policy-options]
policy-statement policy-statement-name {
  dynamic-db;
}
```

You can also define specific policy objects based on the configuration of these objects in the dynamic database. To define a policy object based on the dynamic database,

include the **dynamic-db** statement with the following statements at the **[edit policy-options]** hierarchy level:

- **as-path** *name*
- **as-path-group** *group-name*
- **community** *community-name*
- **condition** *condition-name*
- **prefix-list** *prefix-list-name*

In the standard configuration, you can also define a routing policy that references any policy object you have configured in the standard configuration that references an object configured in the dynamic database.

For example, in standard configuration mode, you configure a prefix list **prefix-list pl2** that references a prefix list, also named **prefix-list pl2**, that has been configured in the dynamic database:

```
[edit policy-options]
prefix-list pl2 {
   dynamic-db; # Reference a prefix list configured in the dynamic database.
}
```

You then configure a routing policy in the standard configuration that includes **prefix-list pl2**:

```
[edit policy-options]
policy-statement one {
   term term1 {
      from {
         prefix-list pl2; # Include the prefix list configured in the standard configuration
            # database, but which references a prefix list configured in the dynamic database.
      }
      then accept;
   }
   then reject;
}
```

If you need to update the configuration of **prefix-list pl2**, you do so in the dynamic database configuration using the **[edit dynamic]** hierarchy level. This enables you to make commit configuration changes to the prefix list more quickly than you can in the standard configuration database.

NOTE: If you are downgrading the Junos OS to Junos OS Release 9.4 or earlier, you must first delete any routing policies that reference the dynamic database. That is, you must delete any routing policies or policy objects configured with the **dynamic-db** statement.

Applying Dynamic Routing Policies to BGP

BGP is the only routing protocol to which you can apply routing policies that reference the dynamic database configuration. You must apply these policies in the standard configuration. Dynamic policies can be applied to BGP export or import policy. They can also be applied at the global, group, or neighbor hierarchy level.

To apply a BGP export policy, include the **export [** *policy-names* **]** statement at the **[edit protocols bgp]**, **[edit protocols bgp group** *group-name***]**, or **[edit protocols bgp group** *group-name* **neighbor** *address***]** hierarchy level.

```
[edit]
protocols
  bgp {
    export [ policy-names ];
  }
}
```

To apply a BGP import policy, include the **import [** *policy-names* **]** statement at the **[edit protocols bgp]**, **[edit protocols bgp group** *group-name***]**, or **[edit protocols bgp group** *group-name* **neighbor** *address***]** hierarchy level.

```
[edit]
protocols
  bgp {
    import [ policy-names ];
  }
}
```

Include one or more policy names configured in that standard configuration at the **[edit policy-options** *policy-statement***]** hierarchy level that reference policies configured in the dynamic database.

Preventing Reestablishment of BGP Peering Sessions After NSR Routing Engine Switchover

If you have active nonstop routing (NSR) enabled, the dynamic database is not synchronized with the backup Routing Engine. As a result, if a switchover to a backup Routing Engine occurs, import and export policies running on the master Routing Engine at the time of the switchover might no longer be available. Therefore, you might want to prevent a BGP peering session from automatically being reestablished as soon as a switchover occurs.

You can configure the router not to reestablish a BGP peering session after an active nonstop routing switchover either for a specified period or until you manually reestablish the session. Include the **idle-after-switch-over (***seconds* **| forever)** statement at the **[edit protocols bgp]**, **[edit protocols bgp group** *group-name***]**, or **[edit protocols bgp group** *group-name* **neighbor** *address***]** hierarchy level:

```
[edit]
bgp {
  protocols {
    idle-after-switch-over (seconds | never);
  }
}
```

For ***seconds***, specify a value from 1 through 4,294,967,295 ($2^{32} - 1$). The BGP peering session is not reestablished until after the specified period. If you specify the **forever** option, the BGP peering session is not established until you issue the **clear bgp neighbor** command.

Related Documentation

- Example: Configuring Dynamic Routing Policies on page 421
- *Junos OS High Availability Library for Routing Devices*

CHAPTER 14

Discard Routing Policy

- Understanding Forwarding Packets to the Discard Interface on page 103

Understanding Forwarding Packets to the Discard Interface

The discard (dsc) interface is not a physical interface, but a virtual interface that discards packets. You can configure one discard interface. This interface allows you to identify the ingress point of a denial-of-service (DoS) attack. When your network is under attack, the target host IP address is identified, and the local policy forwards attacking packets to the discard interface. Traffic routed out of the discard interface is silently discarded.

The discard interface allows you to protect a network from DoS attacks by identifying the target IP address that is being attacked and configuring a policy to forward all packets to a discard interface. All packets forwarded to the discard interface are dropped.

To configure the discard interface, include the **dsc** statement:

```
[edit interfaces interface-name]
dsc {
    unit 0 {
        family inet {
            filter {
                input filter-name;
                output filter-name;
            }
        }
    }
}
```

The **dsc** interface name denotes the discard interface. The discard interface supports only unit 0.

The following two configurations are required to configure a policy to forward all packets to the discard interface.

Configure an input policy to associate a community with the discard interface:

```
[edit]
policy-options {
    community community-name members [ community-id ];
    policy-statement statement-name {
        term term-name {
```

```
        from community community-name;
        then {
          next-hop address; # Remote end of the point-to-point interface
          accept;
        }
      }
    }
  }
```

Configure an output policy to set up the community on the routes injected into the network:

```
[edit]
policy-options {
  policy-statement statement-name {
    term term-name {
      from prefix-list name;
      then community (set | add | delete) community-name;
    }
  }
}
```

Related Documentation

- Example: Forwarding Packets to the Discard Interface on page 435

CHAPTER 15

Reference Tables

- Categories of Routing Policy Match Conditions on page 105
- Routing Policy Match Conditions on page 107
- Route Filter Match Conditions on page 115
- Summary of Routing Policy Actions on page 117
- Actions in Routing Policy Terms on page 119
- Protocol Support for Import and Export Policies on page 130

Categories of Routing Policy Match Conditions

A *match condition* defines the criteria that a route must match. You can define one or more match conditions. If a route matches all match conditions, one or more actions are applied to the route.

Match conditions fall into two categories: standard and extended. In general, the extended match conditions are more complex than standard match conditions. The extended match conditions provide many powerful capabilities. The standard match conditions include criteria that are defined within a routing policy and are less complex than the extended match conditions, also called named match conditions.

Extended match conditions are defined separately from the routing policy and are given names. You then reference the name of the match condition in the definition of the routing policy itself.

Named match conditions allow you to do the following:

- Reuse match conditions in other routing policies.
- Read configurations that include complex match conditions more easily.

Named match conditions include communities, prefix lists, and AS path regular expressions.

Table 18 on page 106 describes each match condition, including its category, when you typically use it, and any relevant notes about it. For more information about match conditions, see "Routing Policy Match Conditions" on page 107.

Table 18: Match Condition Concepts

Match Condition	Category	When to Use	Notes
AS path regular expression—A combination of AS numbers and regular expression operators.	Extended	(BGP only) Match a route based on its AS path. (An AS path consists of the AS numbers of all routers a packet must go through to reach a destination.) You can specify an exact match with a particular AS path or a less precise match.	You use regular expressions to match the AS path.
Community—A group of destinations that share a property. (Community information is included as a path attribute in BGP update messages.)	Extended	Match a group of destinations that share a property. Use a routing policy to define a community that specifies a group of destinations you want to match and one or more actions that you want taken on this community.	Actions can be performed on the entire group. You can create multiple communities associated with a particular destination. You can create match conditions using regular expressions.
Prefix list—A named list of IP addresses.	Extended	Match a route based on prefix information. You can specify an exact match of a particular route only.	You can specify a common action only for all prefixes in the list.
Route list—A list of destination prefixes.	Extended	Match a route based on prefix information. You can specify an exact match of a particular route or a less precise match.	You can specify an action for each prefix in the route list or a common action for all prefixes in the route list.
Standard—A collection of criteria that can match a route.	Standard	Match a route based on one of the following criteria: area ID, color, external route, family, instance (routing), interface name, level number, local preference, metric, neighbor address, next-hop address, origin, preference, protocol, routing table name, or tag. You can specify a match condition for policies based on protocols by naming a protocol from which the route is learned or to which the route is being advertised.	None.
Subroutine—A routing policy that is called repeatedly from another routing policy.	Extended	Use an effective routing policy in other routing policies. You can create a subroutine that you can call over and over from other routing policies.	The subroutine action influences but does not necessarily determine the final action. For more information, see "How a Routing Policy Subroutine Is Evaluated" on page 54.

Each term can consist of two statements, **from** and **to**, that define match conditions:

- In the **from** statement, you define the criteria that an *incoming* route must match. You can specify one or more match conditions. If you specify more than one, all conditions must match the route for a match to occur.

- In the **to** statement, you define the criteria that an *outgoing* route must match. You can specify one or more match conditions. If you specify more than one, all conditions must match the route for a match to occur.

The order of match conditions in a term is not important, because a route must match all match conditions in a term for an action to be taken.

Related Documentation

- Routing Policy Match Conditions on page 107

Routing Policy Match Conditions

Each term in a routing policy can include two statements, **from** and **to**, to define the conditions that a route must match for the policy to apply:

```
from {
  family family-name;
  match-conditions;
  policy subroutine-policy-name;
  prefix-list name;
  route-filter destination-prefix match-type <actions>;
  source-address-filter source-prefix match-type <actions>;
}
to {
  match-conditions;
  policy subroutine-policy-name;
}
```

In the **from** statement, you define the criteria that an incoming route must match. You can specify one or more match conditions. If you specify more than one, they all must match the route for a match to occur.

The **from** statement is optional. If you omit the **from** and the **to** statements, all routes are considered to match.

In export policies, omitting the **from** statement from a routing policy term might lead to unexpected results.

In the **to** statement, you define the criteria that an outgoing route must match. You can specify one or more match conditions. If you specify more than one, they all must match the route for a match to occur. You can specify most of the same match conditions in the **to** statement that you can in the **from** statement. In most cases, specifying a match condition in the **to** statement produces the same result as specifying the same match condition in the **from** statement.

The **to** statement is optional. If you omit both the **to** and the **from** statements, all routes are considered to match.

Table 19 on page 108 summarizes key routing policy match conditions.

Table 19: Summary of Key Routing Policy Match Conditions

Match Condition	Description
aggregate-contributor	Matches routes that are contributing to a configured aggregate. This match condition can be used to suppress a contributor in an aggregate route.
area *area-id*	Matches a route learned from the specified OSPF area during the exporting of OSPF routes into other protocols.
as-path *name*	Matches the name of the path regular expression of an autonomous systems (AS). BGP routes whose AS path matches the regular expression are processed.
color *preference*	Matches a color value. You can specify preference values that are finer-grained than those specified in the ***preference*** match conditions. The **color** value can be a number from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route.
community	Matches the name of one or more communities. If you list more than one name, only one name needs to match for a match to occur. (The matching is effectively a logical OR operation.)
external [type *metric-type***]**	Matches external OSPF routes, including routes exported from one level to another. In this match condition, **type** is an optional keyword. The **metric-type** value can be either 1 or 2. When you do not specify **type**, this condition matches all external routes.
interface *interface-name*	Matches the name or IP address of one or more router interfaces. Use this condition with protocols that are interface-specific. For example, do not use this condition with internal BGP (IBGP). Depending on where the policy is applied, this match condition matches routes learned from or advertised through the specified interface.
internal	Matches a routing policy against the internal flag for simplified next-hop self policies.
level *level*	Matches the IS-IS level. Routes that are from the specified level or are being advertised to the specified level are processed.
local-preference *value*	Matches a BGP local preference attribute. The preference value can be from 0 through 4,294,967,295 ($2^{32} - 1$).
metric *metric* **metric2** *metric*	Matches a metric value. The **metric** value corresponds to the multiple exit discriminator (MED), and **metric2** corresponds to the IGP metric if the BGP next hop runs back through another route.
neighbor *address*	Matches the address of one or more neighbors (peers). For BGP export policies, the address can be for a directly connected or indirectly connected peer. For all other protocols, the address is for the neighbor from which the advertisement is received.
next-hop *address*	Matches the next-hop address or addresses specified in the routing information for a particular route. For BGP routes, matches are performed against each protocol next hop.

Table 19: Summary of Key Routing Policy Match Conditions *(continued)*

Match Condition	Description
origin *value*	Matches the BGP origin attribute, which is the origin of the AS path information. The value can be one of the following: • **egp**—Path information originated from another AS. • **igp**—Path information originated from within the local AS. • **incomplete**—Path information was learned by some other means.
preference *preference* **preference2** *preference*	Matches the preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route.
protocol *protocol*	Matches the name of the protocol from which the route was learned or to which the route is being advertised. It can be one of the following: **aggregate**, **bgp**, **direct**, **dvmrp**, **isis**, **local**, **ospf**, **pim-dense**, **pim-sparse**, **rip**, **ripng**, or **static**.
route-type *value*	Matches the type of route. The value can be either **external** or **internal**.

All conditions in the **from** and **to** statements must match for the action to be taken. The match conditions defined in Table 20 on page 109 are effectively a logical AND operation. Matching in prefix lists and route lists is handled differently. They are effectively a logical OR operation. If you configure a policy that includes some combination of route filters, prefix lists, and source address filters, they are evaluated according to a logical OR operation or a longest-route match lookup.

Table 20 on page 109 describes the match conditions available for matching an incoming or outgoing route. The table indicates whether you can use the match condition in both **from** and **to** statements and whether the match condition functions the same or differently when used with both statements. If a match condition functions differently in a **from** statement than in a **to** statement, or if the condition cannot be used in one type of statement, there is a separate description for each type of statement. Otherwise, the same description applies to both types of statements.

Table 20 on page 109 also indicates whether the match condition is standard or extended. In general, the extended match conditions include criteria that are defined separately from the routing policy (autonomous system [AS] path regular expressions, communities, and prefix lists) and are more complex than standard match conditions. The extended match conditions provide many powerful capabilities. The standard match conditions include criteria that are defined within a routing policy and are less complex than the extended match conditions.

Table 20: Complete List of Routing Policy Match Conditions

Match Condition	Match Condition Category	from Statement Description	to Statement Description
aggregate-contributor	Standard	Match routes that are contributing to a configured aggregate. This match condition can be used to suppress a contributor in an aggregate route.	

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
area *area-id*	Standard	(Open Shortest Path First [OSPF] only) Area identifier. In a **from** statement used with an **export** policy, match a route learned from the specified OSPF area when exporting OSPF routes into other protocols.	
as-path *name*	Extended	(Border Gateway Protocol [BGP] only) Name of an AS path regular expression. For more information, see "Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions" on page 57.	
as-path-group *group-name*	Extended	(BGP only) Name of an AS path group regular expression. For more information, see "Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions" on page 57.	
color *preference* **color2** *preference*	Standard	Color value. You can specify preference values (**color** and **color2**) that are finer-grained than those specified in the **preference** and **preference2** match conditions. The color value can be a number in the range from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route.	
community-count *value* **(equal | orhigher | orlower)**	Standard	(BGP only) Number of community entries required for a route to match. The count value can be a number in the range of 0 through 1,024. Specify one of the following options: • **equal**—The number of communities must equal this value to be considered a match. • **orhigher** —The number of communities must be greater than or equal to this value to be considered a match. • **orlower**—The number of communities must be less than or equal to this value to be considered a match. NOTE: If you configure multiple **community-count** statements, the matching is effectively a logical AND operation. NOTE: The **community-count** attribute only works with standard communities. It does not work with extended communities.	You cannot specify this match condition.
community [*names* **]**	Extended	Name of one or more communities. If you list more than one name, only one name needs to match for a match to occur (the matching is effectively a logical OR operation). For more information, see "Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions" on page 67.	

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
external [type ***metric-type***]	Standard	(OSPF and IS-IS only) Match IGP external routes. For IS-IS routes, the **external** condition also matches routes that are exported from one IS-IS level to another. The **type** keyword is optional and is applicable only to OSPF external routes. When you do not specify **type**, the **external** condition matches all IGP external (OSPF and IS-IS) routes. When you specify **type**, the **external** condition matches only OSPF external routes with the specified OSPF metric type. The metric type can either be 1 or 2. To match BGP external routes, use the **route-type** match condition.	
family ***family-name***	Standard	Name of an address family. Match the address family of the route. Depending on your device and configuration, ***family-name*** can be one of the following: • **inet**— IP version 4 (IPv4) traffic • **inet-mdt**—IPv4 multicast distribution tree (MDT) traffic • **inet-mvpn**—IPv4 multicast virtual private network (MVPN) traffic • **inet-vpn**—IPv4 VPN traffic • **inet6**—IP version 6 (IPv6) traffic • **inet6-mvpn**—IPv6 MVPN traffic • **inet6-vpn**—IPv6 VPN traffic • **iso**—IS-IS traffic • **route-target**—BGP route target filtering routes for VPN traffic Default setting is **inet**.	
instance ***instance-name***	Standard	Name of one or more routing instances. Match a route learned from one of the specified instances.	Name of one or more routing instances. Match a route to be advertised over one of the specified instances.
interface ***interface-name***	Standard	Name or IP address of one or more routing device interfaces. Do not use this qualifier with protocols that are not interface-specific, such as IBGP. Match a route learned from one of the specified interfaces. Direct routes match routes configured on the specified interface.	Name or IP address of one or more routing device interfaces. Do not use this qualifier with protocols that are not interface-specific, such as IBGP. Match a route to be advertised from one of the specified interfaces.
level ***level***	Standard	(Intermediate System-to-Intermediate System [IS-IS] only) IS-IS level. Match a route learned from a specified level.	(IS-IS only) IS-IS level. Match a route to be advertised to a specified level.
local-preference ***value***	Standard	(BGP only) BGP local preference (LOCAL_PREF**local-preference (add | subtract)** ***number***) attribute. The preference value can be a number in the range 0 through 4,294,967,295 ($2^{32} - 1$).	

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
metric *metric* **metric2** *metric* **metric3** *metric* **metric4** *metric*	Standard	Metric value. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. (BGP only) **metric** corresponds to the multiple exit discriminator (MED), and **metric2** corresponds to the interior gateway protocol (IGP) metric if the BGP next hop runs back through another route.	
multicast-scoping (*scoping-name* | *number*) < (**orhigher** | **orlower**) >	Standard	Multicast scope value of IPv4 or IPv6 multicast group address. The multicast-scoping name corresponds to an IPv4 prefix. You can match on a specific multicast-scoping prefix or on a range of prefixes. Specify **orhigher** to match on a scope and numerically higher scopes, or **orlower** to match on a scope and numerically lower scopes. For more information, see the *Multicast Protocols Feature Guide for Routing Devices*. You can apply this scoping policy to the routing table by including the **scope-policy** statement at the **[edit routing-options]** hierarchy level. The ***number*** value can be any hexadecimal number from 0 through F. The multicast-scope value is a number from 0 through 15, or one of the following keywords with the associated meanings: • **node-local** (value=1)—No corresponding prefix • **link-local** (value=2)—Corresponding prefix 224.0.0.0/24 • **site-local** (value=5)—No corresponding prefix • **global** (value=14)—Corresponding prefix 224.0.1.0 through 238.255.255.255 • **organization-local** (value=8)—Corresponding prefix 239.192.0.0/14	
neighbor *address*	Standard	Address of one or more neighbors (peers). For BGP, the address can be a directly connected or indirectly connected peer. For all other protocols, the address is the neighbor from which the advertisement is received. NOTE: The **neighbor** *address* match condition is not valid for the Routing Information Protocol (RIP).	Address of one or more neighbors (peers). For BGP import policies, specifying **to neighbor** produces the same result as specifying **from neighbor**. For BGP export policies, specifying the **neighbor** match condition has no effect and is ignored. For all other protocols, the **to** statement matches the neighbor to which the advertisement is sent. NOTE: The **neighbor** *address* match condition is not valid for the Routing Information Protocol (RIP).
next-hop [*addresses*]	Standard	One or more next-hop addresses specified in the routing information for a particular route. A next-hop address cannot include a netmask. For BGP routes, matches are performed against each protocol next hop.	
next-hop-type merged	Standard	LDP generates a next hop based on RSVP and IP next hops available to use, combined with forwarding-class mapping.	You cannot specify this match condition.

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
nlri-route-type	Standard	Route type from NLRI 1 through NLRI 10. Multiple route types can be specified in a single policy.	
origin ***value***	Standard	(BGP only) BGP origin attribute, which is the origin of the AS path information. The value can be one of the following: • **egp**—Path information originated in another AS. • **igp**—Path information originated within the local AS. • **incomplete**—Path information was learned by some other means.	
policy [***policy-name*** **]**	Extended	Name of a policy to evaluate as a subroutine. For information about this extended match condition, see "Understanding Policy Subroutines in Routing Policy Match Conditions" on page 51.	
preference ***preference*** **preference2** ***preference***	Standard	Preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route. To specify even finer-grained preference values, see the **color** and **color2** match conditions in this table.	
prefix-list ***prefix-list-name*** ***ip-addresses***	Extended	Named list of IP addresses. You can specify an exact match with incoming routes. For information about this extended match condition, see "Understanding Prefix Lists for Use in Routing Policy Match Conditions" on page 45.	You cannot specify this match condition.
prefix-list-filter ***prefix-list-name*** ***match-type***	Extended	Named prefix list. You can specify prefix length qualifiers for the list of prefixes in the prefix list. For information about this extended match condition, see "Understanding Prefix Lists for Use in Routing Policy Match Conditions" on page 45.	You cannot specify this match condition.
protocol ***protocol***	Standard	Name of the protocol from which the route was learned or to which the route is being advertised. It can be one of the following: **access**, **access-internal**, **aggregate**, **arp**, **bgp**, **direct**, **dvmrp**, **esis**, **frr**, **isis**, **l2circuit**, **l2vpn**, **ldp**, **local**, **msdp**, **ospf**, **ospf2**, **ospf3**, **pim**,**rip**, **ripng**, **route-target**, **rsvp**, or **static**. NOTE: The **ospf2** statement matches on OSPFv2 routes. The **ospf3** statement matches on OSPFv3 routes. The **ospf** statement matches on both OSPFv2 and OSPFv3 routes.	

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
rib ***routing-table***	Standard	Name of a routing table. The value of ***routing-table*** can be one of the following: • inet.0—Unicast IPv4 routes • *instance-name* inet.0—Unicast IPv4 routes for a particular routing instance • inet.1—Multicast IPv4 routes • inet.2—Unicast IPv4 routes for multicast reverse-path forwarding (RPF) lookup • inet.3—MPLS routes • mpls.0—MPLS routes for label-switched path (LSP) next hops • inet6.0—Unicast IPv6 routes	
route-filter ***destination-prefix match-type <actions>***	Extended	List of destination prefixes. When specifying a destination prefix, you can specify an exact match with a specific route or a less precise match using match types. You can configure either a common action that applies to the entire list or an action associated with each prefix. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 25.	You cannot specify this match condition.
route-type ***value***	Standard	Type of BGP route. The value can be one of the following: • **external**—External route. • **internal**—Internal route. To match IGP external routes, use the **external** match condition.	
rtf-prefix-list ***name route-targets***	Extended	(BGP only) Named list of route target prefixes for BGP route target filtering and proxy BGP route target filtering. For information about this extended match condition, see *Example: Configuring Proxy BGP Route Target Filtering*.	You cannot specify this match condition.
source-address-filter ***destination-prefix match-type <actions>***	Extended	List of multicast source addresses. When specifying a source address, you can specify an exact match with a specific route or a less precise match using match types. You can configure either a common action that applies to the entire list or an action associated with each prefix. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 25.	You cannot specify this match condition.

Table 20: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
state (active | inactive)	Standard	(BGP export only) Match on the following types of advertised routes: • **active**—An active BGP route • **inactive**—A route advertised to internal BGP peers as the best external path even if the best path is an internal route • **inactive**—A route advertised by BGP as the best route even if the routing table did not select it to be an active route	
tag *string* tag2 *string*	Standard	Tag value. You can specify two tag strings: **tag** (for the first string) and **tag2**. These values are local to the router and can be set on configured routes or by using an import routing policy. You can specify multiple tags under one match condition by including the tags within a bracketed list. For example: **from tag [tag1 tag2 tag3];** For OSPF routes, the**tag** action sets the 32-bit tag field in OSPF external link-state advertisement (LSA) packets. For IS-IS routes, the **tag** action sets the 32-bit flag in the IS-IS IP prefix type length values. (TLV). OSPF stores the INTERNAL route's OSPF area ID in the**tag2** attribute. However, for EXTERNAL routes, OSPF does not store anything in the **tag2**attribute. You can configure a policy term to set the **tag2** value for a route. If the route, already has a **tag2** value (for example, an OSPF route that stores area id in tag2), then the original **tag2** value is overwritten by the new value. When the policy contains the "from area" match condition, for internal OSPF routes, where **tag2** is set, based on the OSPF area- ID, the evaluation is conducted to compare the **tag2** attribute with the area ID. For external OSPF routes that do not have the **tag2** attribute set, the match condition fails.	
validation-database	Standard	When BGP origin validation is configured, triggers a lookup in the route validation database to determine if the route prefix is valid, invalid, or unknown. The route validation database contains route origin authorization (ROA) records that map route prefixes to expected originating autonomous systems (ASs). This prevents the accidental advertisement of invalid routes. See *Example: Configuring Origin Validation for BGP*.	

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 45
- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25

Route Filter Match Conditions

When specifying a destination prefix, you can specify an exact match with a specific route, or a less precise match by using match types. You can configure either a common reject action that applies to the entire list, or an action associated with each prefix.

You can specify known invalid (“bad”) routes to ignore by specifying matches on destination prefixes. Additionally, you can specify that “good” routes be processed in a particular way. For instance, you can group traffic from specific source or destination addresses into forwarding classes to be processed using the class of service (CoS) feature.

Table 21 on page 116 lists route list match types.

Table 21: Route List Match Types

Match Type	Match Conditions
exact	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to the route's prefix length.
longer	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is greater than the route's prefix length.
orlonger	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to or greater than the route's prefix length.
prefix-length-range ***prefix-length2-prefix-length3***	The route shares the same most-significant bits (described by ***prefix-length***), and the route's prefix length falls between ***prefix-length2*** and ***prefix-length3***, inclusive.
through ***destination-prefix***	All the following are true: • The route shares the same most-significant bits (described by ***prefix-length***) of the first destination prefix. • The route shares the same most-significant bits (described by ***prefix-length***) of the second destination prefix for the number of bits in the prefix length. • The number of bits in the route's prefix length is less than or equal to the number of bits in the second prefix. You do not use the **through** match type in most routing policy configurations.
upto ***prefix-length2***	The route shares the same most-significant bits (described by ***prefix-length***) and the route's prefix length falls between ***prefix-length*** and ***prefix-length2***.

Related Documentation

- Categories of Routing Policy Match Conditions on page 105
- Summary of Routing Policy Actions on page 117
- Example: Rejecting Known Invalid Routes on page 218
- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 395

Summary of Routing Policy Actions

An *action* is what the policy framework software does if a route matches all criteria defined in a match condition. You can configure one or more actions in a term.

The policy framework software supports the following types of actions:

- Flow control actions, which affect whether to accept or reject the route or whether to evaluate the next term or routing policy
- Actions that manipulate route characteristics
- Trace action, which logs route matches

Manipulating the route characteristics allows you to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a routing platform's neighbors. You can manipulate the following route characteristics: AS path, class, color, community, damping parameters, destination class, external type, next hop, load balance, local preference, metric, origin, preference, and tag.

For the numeric information (color, local preference, metric, preference, and tag), you can set a specific value or change the value by adding or subtracting a specified amount. The addition and subtraction operations do not allow the value to exceed a maximum value and drop below a minimum value.

All policies have default actions in case one of the following situations arises during policy evaluation:

- A policy does not specify a match condition.
- A match occurs, but a policy does not specify an action.
- A match does not occur with a term in a policy and subsequent terms in the same policy exist.
- A match does not occur by the end of a policy.

An action defines what the router does with the route when the route matches all the match conditions in the **from** and **to** statements for a particular term. If a term does not have **from** and **to** statements, all routes are considered to match and the actions apply to all routes.

Each term can have one or more of the following types of actions. The actions are configured under the **then** statement.

- Flow control actions, which affect whether to accept or reject the route and whether to evaluate the next term or routing policy
- Actions that manipulate route characteristics
- Trace action, which logs route matches

If you do not specify an action, one of the following results occurs:

- The next term in the routing policy, if one exists, is evaluated.
- If the routing policy has no more terms, the next routing policy, if one exists, is evaluated.
- If there are no more terms or routing policies, the accept or reject action specified by the default policy is executed.

Table 22 on page 118 summarizes the routing policy actions.

Table 22: Summary of Key Routing Policy Actions

Action	Description
Flow Control Actions	These actions control the flow of routing information into and out of the routing table.
accept	Accepts the route and propagates it. After a route is accepted, no other terms in the routing policy and no other routing policies are evaluated.
reject	Rejects the route and does not propagate it. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.
next term	Skips to and evaluates the next term in the same routing policy. Any **accept** or **reject** action specified in the **then** statement is ignored. Any actions specified in the **then** statement that manipulate route characteristics are applied to the route.
next policy	Skips to and evaluates the next routing policy. Any **accept** or **reject** action specified in the **then** statement is ignored. Any actions specified in the **then** statement that manipulate route characteristics are applied to the route.
Route Manipulation Actions	These actions manipulate the route characteristics.
as-path-prepend *as-path*	Appends one or more AS numbers at the beginning of the AS path. If you are specifying more than one AS number, include the numbers in quotation marks. The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the appended AS numbers are placed within a confederation sequence. Otherwise, the appended AS numbers are placed with a nonconfederation sequence.
as-path-expand last-as count *n*	Extracts the last AS number in the existing AS path and appends that AS number to the beginning of the AS path *n* times. Replace *n* with a number from **1** through **32**. The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the appended AS numbers are placed within a confederation sequence. Otherwise, the appended AS numbers are placed with a nonconfederation sequence.
class *class-name*	Applies the specified class-of-service (CoS) parameters to routes installed into the routing table.
color *preference* **color2** *preference*	Sets the preference value to the specified value. The **color** and **color2** preference values can be a number from **0** through **4,294,967,295** ($2^{32} - 1$). A lower number indicates a more preferred route.

Table 22: Summary of Key Routing Policy Actions *(continued)*

Action	Description
damping *name*	Applies the specified route-damping parameters to the route. These parameters override BGP's default damping parameters. This action is useful only in import policies.
local-preference *value*	Sets the BGP local preference attribute. The preference can be a number from **0** through **4,294,967,295** ($2^{32}-1$).
metric *metric* **metric2** *metric* **metric3** *metric* **metric4** *metric*	Sets the metric. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. For BGP routes, **metric** corresponds to the MED, and **metric2** corresponds to the IGP metric if the BGP next hop loops through another router.
next-hop *address*	Sets the next hop. If you specify ***address*** as **self**, the next-hop address is replaced by one of the local router's addresses. The advertising protocol determines which address to use.

Related Documentation

- *Firewall Filters Feature Guide for Routing Devices*
- *Traffic Policers Feature Guide for Routing Devices*

Actions in Routing Policy Terms

Each term in a routing policy can include a **then** statement, which defines the actions to take if a route matches all the conditions in the **from** and **to** statements in the term:

```
then {
  actions;
}
```

You can include this statement at the following hierarchy levels:

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

If a term does not have **from** and **to** statements, all routes are considered to match, and the actions apply to them all. For information about the **from** and **to** statements, see "Routing Policy Match Conditions" on page 107.

You can specify one or more actions in the **then** statement. There are three types of actions:

- Flow control actions, which affect whether to accept or reject the route and whether to evaluate the next term or routing policy.

- Actions that manipulate route characteristics.
- Trace action, which logs route matches.

NOTE: When you specify an action that manipulates the route characteristics, the changes occur in a copy of the source route. The source route itself does not change. The effect of the action is visible only after the route is imported into or exported from the routing table. To view the source route before the routing policy has been applied, use the **show route receive-protocol** command. To view a route after an export policy has been applied, use the **show route advertised-protocol** command.

During policy evaluation, the characteristics in the copy of the source route always change immediately after the action is evaluated. However, the route is not copied to the routing table or a routing protocol until the completion of the policy evaluation is complete.

The **then** statement is optional. If you omit it, one of the following occurs:

- The next term in the routing policy, if one is present, is evaluated.
- If there are no more terms in the routing policy, the next routing policy, if one is present, is evaluated.
- If there are no more terms or routing policies, the accept or reject action specified by the default policy is taken. For more information, see "Default Routing Policies" on page 15.

The following sections discuss the following actions:

- Configuring Flow Control Actions on page 120
- Configuring Actions That Manipulate Route Characteristics on page 121
- Configuring the Default Action in Routing Policies on page 127
- Configuring a Final Action in Routing Policies on page 128
- Logging Matches to a Routing Policy Term on page 129
- Configuring Separate Actions for Routes in Route Lists on page 129

Configuring Flow Control Actions

Table 23 on page 121 lists the flow control actions. You can specify one of these actions along with the trace action or one or more of the actions that manipulate route characteristics (see "Configuring Actions That Manipulate Route Characteristics" on page 121).

Table 23: Flow Control Actions

Flow Control Action	Description
accept	Accept the route and propagate it. After a route is accepted, no other terms in the routing policy and no other routing policies are evaluated.
default-action accept	Accept and override any action intrinsic to the protocol. This is a nonterminating policy action.
reject	Reject the route and do not propagate it. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.
default-action reject	Reject and override any action intrinsic to the protocol. This is a nonterminating policy action.
next term	Skip to and evaluate the next term in the same routing policy. Any accept or reject action specified in the **then** statement is skipped. Any actions in the **then** statement that manipulate route characteristics are applied to the route. **next term** is the default control action if a match occurs and you do not specify a flow control action.
next policy	Skip to and evaluate the next routing policy. Any accept or reject action specified in the **then** statement is skipped. Any actions in the **then** statement that manipulate route characteristics are applied to the route. **next policy** is the default control action if a match occurs, you do not specify a flow control action, and there are no further terms in the current routing policy.

Configuring Actions That Manipulate Route Characteristics

You can specify one or more of the actions listed in Table 24 on page 121 to manipulate route characteristics.

Table 24: Actions That Manipulate Route Characteristics

Action	Description
as-path-prepend *as-path*	(BGP only) Affix one or more AS numbers at the beginning of the AS path. If specifying more than one AS number, enclose the numbers in quotation marks (" "). The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the affixed AS numbers are placed within a confederation sequence. Otherwise, the affixed AS numbers are placed with a nonconfederation sequence. For more information, see "Understanding Prepending AS Numbers to BGP AS Paths" on page 63. In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS.

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
as-path-expand last-as count ***n***	(BGP only) Extract the last AS number in the existing AS path and affix that AS number to the beginning of the AS path ***n*** times, where ***n*** is a number from 1 through 32. The AS number is added before the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the affixed AS numbers are placed within a confederation sequence. Otherwise, the affixed AS numbers are placed within a nonconfederation sequence. This option is typically used in non-IBGP export policies.
class ***class-name***	(Class of service [CoS] only) Apply the specified class-of-service parameters to routes installed into the routing table. For more information, see the *Junos OS Class of Service Library for Routing Devices*.
color ***preference*** **color2** ***preference***	Set the preference value to the specified value. The **color** and **color2** preference values are even more fine-grained than those specified in the **preference** and **preference2** actions. The color value can be a number in the range from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route. If you set the preference with the **color** action, the value is internal to the Junos OS and is not transitive.
color (add | subtract) ***number*** **color2 (add | subtract)** ***number***	Change the color preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} - 1$), the value is set to $2^{32} - 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
community (+ | add) [***names*** **]**	(BGP only) Add the specified communities to the set of communities in the route. For more information, see "Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions" on page 67.
community (– | delete) [***names*** **]**	(BGP only) Delete the specified communities from the set of communities in the route. For more information, see "Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions" on page 67.
community (= | set) [***names*** **]**	(BGP only) Replace any communities that were in the route in with the specified communities. For more information, see "Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions" on page 67.
cos-next-hop-map ***map-name***	Set CoS-based next-hop map in forwarding table.
damping ***name***	(BGP only) Apply the specified route-damping parameters to the route. These parameters override the default damping parameters. This action is useful only in an import policy, because the damping parameters affect the state of routes in the routing table. To apply damping parameters, you must enable BGP flap damping as described in the *Junos OS Routing Protocols Library for Routing Devices*, and you must create a named list of parameters as described in "Using Routing Policies to Damp BGP Route Flapping" on page 82.

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
destination-class ***destination-class-name***	Maintain packet counts for a route passing through your network, based on the destination address in the packet. You can do the following: • Configure group destination prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding destination class. • Enable packet counting on one or more interfaces by including the **destination-class-usage** statement at the **[edit interfaces** ***interface-name*** **unit** ***logical-unit-number*** **family inet accounting]** hierarchy level (see the *Junos OS Class of Service Library for Routing Devices*). • View the output by using one of the following commands: **show interfaces destination-class (all |** ***destination-class-name logical-interface-name***), **show interfaces** ***interface-name*** **extensive**, or **show interfaces** ***interface-name*** **statistics** (see the *Junos OS Operational Mode Commands*). • To configure a packet count based on the source address, use the **source-class** statement described in this table.
external type ***metric***	Set the external metric type for routes exported by OSPF. You must specify the keyword **type**.
forwarding-class ***forwarding-class-name***	Create the forwarding class that includes packets based on both the destination address and the source address in the packet. You can do the following: • Configure group prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding forwarding class. • Enable packet counting on one or more interfaces by using the procedure described in either the **destination-class** or **source-class** actions defined in this table.
install-nexthop <strict> lsp ***lsp-name***	Choose which next hops, among a set of equal LSP next hops, are installed in the forwarding table. Use the export policy for the forwarding table to specify the LSP next hop to be used for the desired routes. Specify the **strict** option to enable strict mode, which checks to see if any of the LSP next hops specified in the policy are up. If none of the specified LSP next hops are up, the policy installs the discard next hop.
install-to-fib	For PTX Series routers only, override the default BGP routing policy. For more information, see *Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers*.
load-balance per-packet	(For export to the forwarding table only) Install all next-hop addresses in the forwarding table and have the forwarding table perform per-packet load balancing. This policy action allows you to optimize VPLS traffic flows across multiple paths. For more information, see *Configuring Per-Packet Load Balancing*.
load-balance per-prefix	For PTX Series routers only, override the default per-packet load balancing routing policy for BGP. For more information, see *Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers*.
local-preference ***value***	(BGP only) Set the BGP local preference (LOCAL_PREF) attribute. The preference value can be a number in the range from 0 through 4,294,967,295 ($2^{32} - 1$).

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
local-preference (add | subtract) ***number***	Change the local preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} - 1$), the value is set to $2^{32} - 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value. For BGP, if the attribute value is not known, it is initialized to 100 before the routing policy is applied.
map-to-interface (*interface-name* | self)	Sets the **map-to-interface** value which is similar to existing metric or tag actions. The **map-to-interface** action requires you to specify one of the following: • A logical interface (for example, ge-0/0/0.0). The logical interface can be any interface that multicast currently supports, including VLAN and aggregated Ethernet interfaces. NOTE: If you specify a physical interface as the **map-to-interface** (for example, ge-0/0/0), a value of .0 is appended to physical interface to create a logical interface. • The keyword **self**. The **self** keyword specifies that multicast data packets are sent on the same interface as the control packets and no mapping occurs. If no term matches, then no multicast data packets are sent.
metric ***metric*** **metric2** ***metric*** **metric3** ***metric*** **metric4** ***metric***	Set the metric. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. (BGP only) **metric** corresponds to the MED, and **metric2** corresponds to the IGP metric if the BGP next hop loops through another router.
metric (add | subtract) ***number*** **metric2 (add | subtract)** ***number*** **metric3 (add | subtract)** ***number*** **metric4 (add | subtract)** ***number***	Change the metric value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} - 1$), the value is set to $2^{32} - 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
metric expression (metric multiplier ***x*** **offset** ***a*** **| metric2 multiplier** ***y*** **offset** ***b***)	Calculate a metric based on the current values of **metric** and **metric2**. This policy action overrides the current value of the metric attribute with the result of the expression ((x * metric) + a) + ((y * metric2) + b) where **metric** and **metric2** are the current input values. Metric multipliers are limited in range to eight significant digits.
metric (igp | minimum-igp) ***site-offset***	(BGP only) Change the metric (MED) value by the specified negative or positive offset. This action is useful only in an external BGP (EBGP) export policy.

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description

next-hop (*address* | discard | next-table *table-name* | peer-address | reject | self)	Set the next-hop address. When the advertising protocol is BGP, you can set the next hop only when any third-party next hop can be advertised; that is, when you are using IBGP or EBGP confederations.
	
	If you specify **self**, the next-hop address is replaced by one of the local routing device's addresses. The advertising protocol determines which address to use. When the advertising protocol is BGP, this address is set to the local IP address used for the BGP adjacency. A routing device cannot install routes with itself as the next hop.
	
	If you specify **peer-address**, the next-hop address is replaced by the peer's IP address. This option is valid only in import policies. Primarily used by BGP to enforce using the peer's IP address for advertised routes, this option is meaningful only when the next hop is the advertising routing device or another directly connected routing device.
	
	If you specify **discard**, the next-hop address is replaced by a discard next hop.
	
	If you specify **next-table**, the routing device performs a forwarding lookup in the specified table.
	
	If you use the **next-table** action, the configuration must include a term qualifier that specifies a different table than the one specified in the **next-table** action. In other words, the term qualifier in the **from** statement must exclude the table in the **next-table** action. In the following example, the first term contains **rib vrf-customer2.inet.0** as a matching condition. The action specifies a next-hop in a different routing table, **vrf-customer1.inet.0**. The second term does the opposite by using **rib vrf-customer1.inet.0** in the match condition and **vrf-customer2.inet.0** In the **next-table** action.
	
	`term 1 {`
	`  from {`
	`  protocol bgp;`
	`  rib vrf-customer2.inet.0;`
	`  community customer;`
	`  }`
	`  then {`
	`  next-hop next-table vrf-customer1.inet.0;`
	`  }`
	`}`
	`term 2 {`
	`  from {`
	`  protocol bgp;`
	`  rib vrf-customer1.inet.0;`
	`  community customer;`
	`  }`
	`  then {`
	`  next-hop next-table vrf-customer2.inet.0;`
	`  }`
	`}`
	
	If you specify **reject**, the next-hop address is replaced by a reject next hop.

origin *value*	(BGP only) Set the BGP origin attribute to one of the following values: • **igp**—Path information originated within the local AS. • **egp**—Path information originated in another AS. • **incomplete**—Path information learned by some other means.

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
p2mp-lsp-root	Set the ingress root node for a multipoint LDP (M-LDP)-based point-to-multipoint label-switched path (LSP). For more information, see *Example: Configuring Multipoint LDP In-Band Signaling for Point-to-Multipoint LSPs*.
preference *preference* **preference2** *preference*	Set the preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number in the range from 0 through 4,294,967,295 ($2^{32} - 1$). A lower number indicates a more preferred route. To specify even finer-grained preference values, see the **color** and **color2** actions in this table. If you set the preference with the **preference** action, the new preference remains associated with the route. The new preference is internal to the Junos OS and is not transitive.
preference (add | subtract) *number* **preference2 (add | subtract)** *number*	Change the preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} - 1$), the value is set to $2^{32} - 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
priority (low | medium | high)	(OSPF import only) Specify a priority for prefixes included in an OSPF import policy. Prefixes learned through OSPF are installed in the routing table based on the priority assigned to the prefixes. Prefixes assigned a priority of **high** are installed first, while prefixes assigned a priority of **low** are installed last. NOTE: An OSPF import policy can only be used to set priority or to filter OSPF external routes. If an OSPF import policy is applied that results in a **reject** terminating action for a nonexternal route, then the **reject** action is ignored and the route is accepted anyway.
source-class *source-class-name*	Maintain packet counts for a route passing through your network, based on the source address. You can do the following: • Configure group source prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding source class. • Enable packet counting on one or more interfaces by including the **source-class-usage** *interface-name* statement at the **[edit interfaces** *logical-unit-number* **unit family inet accounting]** hierarchy level. Also, follow the **source-class-usage** statement with the **input** or **output** statement to define the inbound and outbound interfaces on which traffic monitored for source-class usage (SCU) is arriving and departing (or define one interface for both). The complete syntax is **[edit interfaces** *interface-name* **unit family inet accounting source-class-usage (input | output | input output)** *unit-number***]**. • View the output by using one of the following commands: **show interfaces** *interface-name* **source-class** *source-class-name*, **show interfaces** *interface-name* **extensive**, or **show interfaces** *interface-name* **statistics** (see the *Junos OS Operational Mode Commands*). • To configure a packet count based on the destination address, use the **destination-class** statement described in this table. • For a detailed source-class usage example configuration, see the *Source Class Usage Feature Guide*. NOTE: When configuring policy action statements, you can configure only one source class for each matching route. In other words, more than one source class cannot be applied to the same route.

Table 24: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
ssm-source [***addresses*** **];**	Specify one or more IPv4 or IPv6 source addresses for the source-specific multicast (SSM) policy
ssm-source [***addresses*** **];**	Specify one or more IPv4 or IPv6 source addresses for the source-specific multicast (SSM) policy.
tag *tag* **tag2** *tag*	Set the tag value. You can specify two tag strings: **tag** (for the first string) and **tag2** (a second string). These values are local to the router. • For OSPF routes the **tag** action sets the 32-bit tag field in OSPF external link-state advertisement (LSA) packets. • For IS-IS routes, the **tag** action sets the 32-bit flag in the IS-IS IP prefix type length values (TLV). • For RIPv2 routes, the **tag** action sets the route-tag community. The **tag2** option is not supported.
tag (add | subtract) *number* **tag2 (add | subtract)** *number*	Change the tag value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} - 1$), the value is set to $2^{32} - 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
validation-state	When BGP origin validation is configured, set the validation state of a route prefix to valid, invalid, or unknown. The route validation database contains route origin authorization (ROA) records that map route prefixes to expected originating autonomous systems (ASs). This prevents the accidental advertisement of invalid routes. See *Example: Configuring Origin Validation for BGP*.

Configuring the Default Action in Routing Policies

The **default-action** statement overrides any action intrinsic to the protocol. This action is also nonterminating, so that various policy terms can be evaluated before the policy is terminated. You can specify a default action, either **accept** or **reject**, as follows:

```
[edit]
policy-options {
  policy-statement policy-name {
    term term-name {
      from {
        family family-name;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
```

```
        policy subroutine-policy-name;
      }
      then {
        actions;
        default-action (accept | reject);
      }
    }
  }
}
```

The resulting action is set either by the protocol or by the last policy term that is matched.

Example: Configuring the Default Action in a Routing Policy

Configure a routing policy that matches routes based on three policy terms. If the route matches the first term, a certain community tag is attached. If the route matches two separate terms, then both community tags are attached. If the route does not match any terms, it is rejected (protocol's default action). Note that the terms **hub** and **spoke** are mutually exclusive.

```
[edit]
policy-options {
  policy-statement test {
    term set-default {
      then default-action reject;
    }
    term hub {
      from interface ge-2/1/0.5;
      then {
        community add test-01-hub;
        default-action accept;
      }
    }
    term spoke {
      from interface [ ge-2/1/0.1 ge-2/1/0.2 ];
      then {
        community add test-01-spoke;
        default-action accept;
      }
    }
    term management {
      from protocol direct;
      then {
        community add management;
        default-action accept;
      }
    }
  }
}
```

Configuring a Final Action in Routing Policies

In addition to specifying an action using the **then** statement in a named term, you can also specify an action using the **then** statement in an unnamed term, as follows:

```
[edit]
```

```
policy-options {
  policy-statement policy-name {
    term term-name {
      from {
        family family-name;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
        policy subroutine-policy-name;
      }
      then {
        actions;
      }
    }
    then action;
  }
}
```

Logging Matches to a Routing Policy Term

If you specify the trace action, the match is logged to a trace file. To set up a trace file, you must specify the following elements in the global **traceoptions** statement:

- Trace filename
- **policy** option in the **flag** statement

The following example uses the trace filename of **policy-log**:

```
[edit]
routing-options {
  traceoptions {
    file "policy-log";
    flag policy;
  }
}
```

This action does not affect the flow control during routing policy evaluation.

If a term that specifies a trace action also specifies a flow control action, the name of the term is logged in the trace file. If a term specifies a trace action only, the word <**default**> is logged.

Configuring Separate Actions for Routes in Route Lists

If you specify route lists in the **from** statement, for each route in the list, you can specify an action to take on that individual route directly, without including a **then** statement. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 25.

Related Documentation

- Route Filter Match Conditions on page 115
- Routing Policy Match Conditions on page 107

Protocol Support for Import and Export Policies

Table 25: Protocol Support for Import and Export Policies

Protocol	Import Policy	Export Policy	Supported Levels
BGP	Yes	Yes	Import: global, group, peer Export: global, group, peer
DVMRP	Yes	Yes	Global
IS-IS	No	Yes	Export: global
LDP	Yes	Yes	Global
MPLS	No	No	–
OSPF	Yes	Yes	Export: global Import: external routes only
PIM dense mode	Yes	Yes	Global
PIM sparse mode	Yes	Yes	Global
Pseudoprotocol—Explicitly configured routes, which include the following: • Aggregate routes • Generated routes	Yes	No	Import: global
RIP and RIPng	Yes	Yes	Import: global, neighbor Export: group

PART 2

Configuration

- Routing Policy Evaluation on page 133
- Route Filters on page 229
- Prefix Lists on page 261
- Subroutines on page 273
- AS Paths on page 283
- Communities on page 327
- Testing Policies on page 367
- Damp BGP Route Flapping on page 375
- Source Class Usage and Destination Class Usage on page 395
- Conditional Routing Policies on page 405
- Dynamic Routing Policies on page 421
- Discard Routing Policy on page 435
- Routing Policy Configuration Statements on page 445

CHAPTER 16

Routing Policy Evaluation

- Example: Applying Routing Policies at Different Levels of the BGP Hierarchy on page 133
- Example: Configuring a Conditional Default Route Policy on page 142
- Example: Using Routing Policy in an ISP Network on page 149
- Example: Disabling Suppression of Route Advertisements on page 197
- Example: Configuring BGP to Advertise the Best External Route to Internal Peers on page 204
- Example: Setting BGP to Advertise Inactive Routes on page 212
- Example: Rejecting Known Invalid Routes on page 218
- Example: Using Routing Policy to Set a Preference Value for BGP Routes on page 223

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

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 on page 133
- Overview on page 133
- Configuration on page 135
- Verification on page 139

Requirements

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

Overview

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).

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 1.1.1.1;
   export send-direct;
   group internal-peers {
      type internal;
      export send-192.168.0.1;
      neighbor 2.2.2.2 {
         export send-192.168.20.1;
      }
      neighbor 3.3.3.3;
   }
   group other-group {
      type internal;
      neighbor 4.4.4.4;
   }
}
```

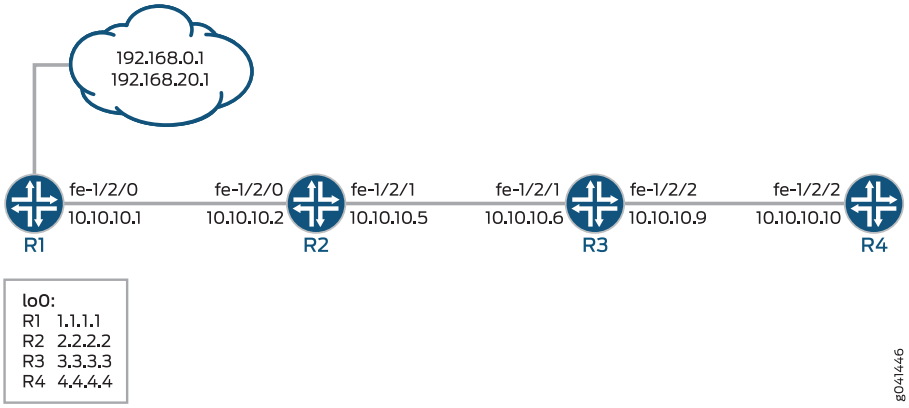
A key point, and one that is often misunderstood and that can lead to problems, is that in such a configuration, only the most explicit policy is applied. A neighbor-level policy is more explicit than a group-level policy, which in turn is more explicit than a global policy.

The neighbor 2.2.2.2 is subjected only to the send-192.168.20.1 policy. The neighbor 3.3.3.3, lacking anything more specific, is subjected only to the send-192.168.0.1 policy. Meanwhile, neighbor 4.4.4.4 in group other-group has no group or neighbor-level policy, so it uses the send-direct policy.

If you need to have neighbor 2.2.2.2 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 2.2.2.2.

Figure 17 on page 135 shows the sample network.

Figure 17: Applying Routing Policies to BGP



"CLI Quick Configuration" on page 135 shows the configuration for all of the devices in Figure 17 on page 135.

The section "Step-by-Step Procedure" on page 136 describes the steps on Device R1.

Configuration

CLI Quick Configuration

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

Device 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 1.1.1.1/32
set protocols bgp local-address 1.1.1.1
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 2.2.2.2 export send-static-192.168.20
set protocols bgp group internal-peers neighbor 3.3.3.3
set protocols bgp group other-group type internal
set protocols bgp group other-group neighbor 4.4.4.4
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 1.1.1.1
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 2.2.2.2/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 2.2.2.2
set protocols bgp group internal-peers neighbor 3.3.3.3
set protocols bgp group internal-peers neighbor 1.1.1.1
set protocols bgp group internal-peers neighbor 4.4.4.4
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 2.2.2.2
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 3.3.3.3/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 3.3.3.3
set protocols bgp group internal-peers neighbor 2.2.2.2
set protocols bgp group internal-peers neighbor 1.1.1.1
set protocols bgp group internal-peers neighbor 4.4.4.4
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 3.3.3.3
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 4.4.4.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 4.4.4.4
set protocols bgp group internal-peers neighbor 2.2.2.2
set protocols bgp group internal-peers neighbor 1.1.1.1
set protocols bgp group internal-peers neighbor 3.3.3.3
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 4.4.4.4
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 *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 1.1.1.1/32
```

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 1.1.1.1
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 2.2.2.2 export send-static-192.168.20
user@R1# set group internal-peers neighbor 3.3.3.3
user@R1# set group other-group type internal
user@R1# set group other-group neighbor 4.4.4.4
```

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

```
[edit routing-options]
user@R1# set router-id 1.1.1.1
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 1.1.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    local-address 1.1.1.1;
    export send-direct;
    group internal-peers {
        type internal;
        export send-static-192.168.0;
        neighbor 2.2.2.2 {
            export send-static-192.168.20;
        }
        neighbor 3.3.3.3;
    }
    group other-group {
        type internal;
        neighbor 4.4.4.4;
    }
}
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 1.1.1.1;
autonomous-system 17;
```

Verification

Confirm that the configuration is working properly.

- Verifying BGP Route Learning on page 139
- Verifying BGP Route Receiving on page 141

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

1.1.1.1/32         *[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 1.1.1.1
                      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 1.1.1.1
                      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

1.1.1.1/32          [BGP/170] 1d 20:38:54, localpref 100, from 1.1.1.1
                      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 1.1.1.1
                      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: 1.1.1.1/32 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 1.1.1.1/32 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 1.1.1.1

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.20.1/32         1.1.1.1                      100        I

user@R3> show route receive-protocol bgp 1.1.1.1

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.0.1/32          1.1.1.1                      100        I

user@R4> show route receive-protocol bgp 1.1.1.1

inet.0: 9 destinations, 11 routes (9 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  1.1.1.1/32              1.1.1.1                      100        I
  10.10.10.0/30           1.1.1.1                      100        I
```

Meaning On Device R2, the **route receive-protocol bgp 1.1.1.1** 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 1.1.1.1** 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 1.1.1.1** command shows that Device R4 received two BGP routes, 1.1.1.1/32 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.

Related Documentation

- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring a Policy Subroutine on page 273
- Example: Configuring Routing Policy Prefix Lists on page 261
- export on page 458
- import on page 467

Example: Configuring a Conditional Default Route Policy

This example shows how to configure a conditional default route on one routing device and redistribute the default route into OSPF.

- Requirements on page 142
- Overview on page 142
- Configuration on page 143
- Verification on page 147

Requirements

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

Overview

In this example, OSPF area 0 contains three routing devices. Device R3 has a BGP session with an external peer, for example, an Internet service provider (ISP).

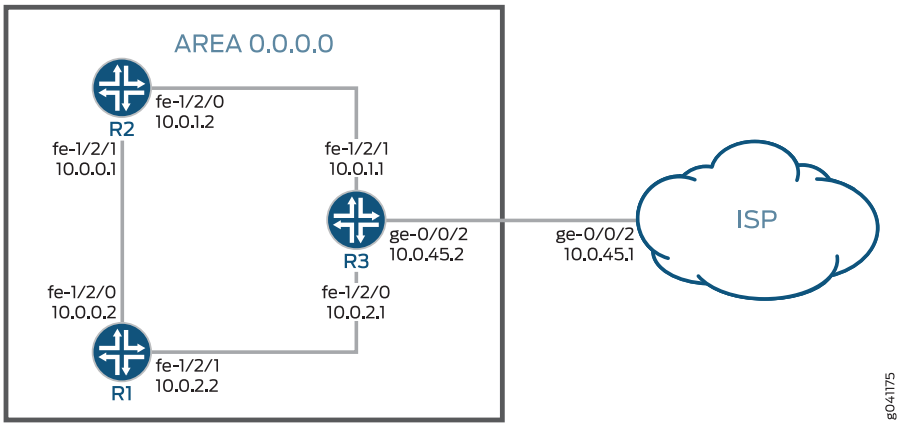
To propagate a static route into BGP, this example includes the **discard** statement when defining the route. The ISP injects a default static route into BGP, which provides the customer network with a default static route to reach external networks. The static route has a discard next hop. This means that if a packet does not match a more specific route, the packet is rejected and a reject route for this destination is installed in the routing table, but Internet Control Message Protocol (ICMP) unreachable messages are not sent. The discard next hop allows you to originate a summary route, which can be advertised through dynamic routing protocols.

Device R3 exports the default route into OSPF. The route policy on Device R3 is conditional such that if the connection to the ISP goes down, the default route is no longer exported into OSPF because it is no longer active in the routing table. This policy prevents packets from being silently dropped without notification (also known as black holing).

This example shows the configuration for all of the devices and the step-by-step configuration on Device R3.

Figure 18 on page 143 shows the sample network.

Figure 18: OSPF with a Conditional Default Route to an ISP



Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 0 description R1->R3
set interfaces fe-1/2/0 unit 0 family inet address 10.0.1.2/30
set interfaces fe-1/2/1 unit 2 description R1->R2
set interfaces fe-1/2/1 unit 2 family inet address 10.0.0.1/30
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
```

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 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
```

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->R1
set interfaces fe-1/2/1 unit 5 family inet address 10.0.1.1/30
set interfaces ge-0/0/2 unit 0 description R3->ISP
set interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30
set protocols bgp group ext type external
set protocols bgp group ext peer-as 65000
set protocols bgp group ext neighbor 10.0.45.1
set protocols ospf export gendefault
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set policy-options policy-statement gendefault term upstreamroutes from protocol bgp
set policy-options policy-statement gendefault term upstreamroutes from as-path
  upstream
set policy-options policy-statement gendefault term upstreamroutes from route-filter
  0.0.0.0/0 upto /16
```

```
set policy-options policy-statement gendefault term upstreamroutes then next-hop
  10.0.45.1
set policy-options policy-statement gendefault term upstreamroutes then accept
set policy-options policy-statement gendefault term end then reject
set policy-options as-path upstream "^65000 "
set routing-options generate route 0.0.0.0/0 policy gendefault
set routing-options autonomous-system 65001
```

Device ISP

```
set interfaces ge-0/0/2 unit 0 family inet address 10.0.45.1/30
set protocols bgp group ext type external
set protocols bgp group ext export advertise-default
set protocols bgp group ext peer-as 65001
set protocols bgp group ext neighbor 10.0.45.2
set policy-options policy-statement advertise-default term 1 from route-filter 0.0.0.0/0
  exact
set policy-options policy-statement advertise-default term 1 then accept
set routing-options static route 0.0.0.0/0 discard
set routing-options autonomous-system 65000
```

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 R3:

1. Configure the device interfaces.

```
[edit interfaces]
user@R3# set fe-1/2/0 unit 3 description R3->R2
user@R3# set fe-1/2/0 unit 3 family inet address 10.0.2.1/30

user@R3# set fe-1/2/1 unit 5 description R3->R1
user@R3# set fe-1/2/1 unit 5 family inet address 10.0.1.1/30

user@R3# set ge-0/0/2 unit 0 description R3->ISP
user@R3# set ge-0/0/2 unit 0 family inet address 10.0.45.2/30
```

2. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R3# set autonomous-system 65001
```

3. Configure the BGP session with the ISP device.

```
[edit protocols bgp group ext]
user@R3# set type external
user@R3# set peer-as 65000
user@R3# set neighbor 10.0.45.1
```

4. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R3# set interface fe-1/2/1.4
user@R3# set interface fe-1/2/0.3
```

5. Configure the routing policy.

```
[edit policy-options policy-statement gendefault]
user@R3# set term upstreamroutes from protocol bgp
user@R3# set term upstreamroutes from as-path upstream
user@R3# set term upstreamroutes from route-filter 0.0.0.0/0 upto /16
user@R3# set term upstreamroutes then next-hop 10.0.45.1
user@R3# set term upstreamroutes then accept

user@R3# set term end then reject

[edit policy-options]
user@R3# set as-path upstream "^65000 "
```

6. Configure the generated route, associating the routing policy with the generated route.

```
[edit routing-options]
user@R3# set generate route 0.0.0.0/0 policy gendefault
```

7. Apply the export policy to OSPF.

```
[edit protocols ospf]
user@R3# set export gendefault
```

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

```
[edit]
user@R3# commit
```

Results

Confirm your configuration by issuing the **show** command. 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 3 {
            description R3->R2;
            family inet {
                address 10.0.2.1/30;
            }
        }
    }
    fe-1/2/1 {
        unit 5 {
            description R3->R1;
            family inet {
                address 10.0.1.1/30;
            }
        }
    }
    ge-1/2/0 {
        unit 0 {
```

```
        description R3->ISP;
        family inet {
          address 10.0.45.2/30;
        }
      }
    }
}
protocols {
    bgp {
      group ext {
        type external;
        peer-as 65000;
        neighbor 10.0.45.1;
      }
    }
    ospf {
      export gendefault;
      area 0.0.0.0 {
        interface fe-1/2/1.4;
        interface fe-1/2/0.3;
      }
    }
}
policy-options {
    policy-statement gendefault {
      term upstreamroutes {
        from {
          protocol bgp;
          as-path upstream;
          route-filter 0.0.0.0/0 upto /16;
        }
        then {
          next-hop 10.0.45.1;
          accept;
        }
      }
      term end {
        then reject;
      }
    }
    as-path upstream "^65000 ";
}
routing-options {
    generate {
      route 0.0.0.0/0 policy gendefault;
    }
    autonomous-system 65001;
}
```

Verification

Confirm that the configuration is working properly.

- Verifying That the Route to the ISP Is Working on page 147
- Verifying That the Static Route Is Redistributed on page 147
- Testing the Policy Condition on page 148

Verifying That the Route to the ISP Is Working

Purpose Make sure connectivity is established between Device R3 and the ISP's router.

Action

```
user@R3> ping 10.0.45.1
PING 10.0.45.1 (10.0.45.1): 56 data bytes
64 bytes from 10.0.45.1: icmp_seq=0 ttl=64 time=1.185 ms
64 bytes from 10.0.45.1: icmp_seq=1 ttl=64 time=1.199 ms
64 bytes from 10.0.45.1: icmp_seq=2 ttl=64 time=1.186 ms
```

Meaning The **ping** command confirms reachability.

Verifying That the Static Route Is Redistributed

Purpose Make sure that the BGP policy is redistributing the static route into Device R3's routing table. Also make sure that the OSPF policy is redistributing the static route into the routing tables of Device R1 and Device R2.

Action

```
user@R3> show route protocol bgp

inet.0: 9 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:00:25, localpref 100
                      AS path: 65000 I
                    > to 10.0.45.1 via ge-0/0/2.6

user@R1> show route protocol ospf

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/150] 00:03:58, metric 0, tag 0
                    > to 10.0.1.1 via fe-1/2/0.0
10.0.2.0/30        *[OSPF/10] 03:37:45, metric 2
                      to 10.0.1.1 via fe-1/2/0.0
                    > to 10.0.0.2 via fe-1/2/1.2
224.0.0.5/32       *[OSPF/10] 03:38:41, metric 1
                      MultiRecv

user@R2> show route protocol ospf
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/150] 00:04:04, metric 0, tag 0
                    > to 10.0.2.1 via fe-1/2/1.4
10.0.1.0/30        *[OSPF/10] 03:37:46, metric 2
                      to 10.0.0.1 via fe-1/2/0.1
                    > to 10.0.2.1 via fe-1/2/1.4
224.0.0.5/32       *[OSPF/10] 03:38:47, metric 1
                      MultiRecv
```

Meaning The routing tables contain the default 0.0.0.0/0 route. If Device R1 and Device R2 receive packets destined for networks not specified in their routing tables, those packets will be sent to Device R3 for further processing. If Device R3 receives packets destined for networks not specified in its routing table, those packets will be sent to the ISP for further processing.

Testing the Policy Condition

Purpose Deactivate the interface to make sure that the route is removed from the routing tables if the external network becomes unreachable.

Action

```
user@R3> deactivate interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30
user@R3> commit

user@R1> show route protocol ospf

inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.2.0/30        *[OSPF/10] 03:41:48, metric 2
                      to 10.0.1.1 via fe-1/2/0.0
                    > to 10.0.0.2 via fe-1/2/1.2
224.0.0.5/32       *[OSPF/10] 03:42:44, metric 1
                      MultiRecv

user@R2> show route protocol ospf
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.1.0/30        *[OSPF/10] 03:42:10, metric 2
                      to 10.0.0.1 via fe-1/2/0.1
                    > to 10.0.2.1 via fe-1/2/1.4
224.0.0.5/32       *[OSPF/10] 03:43:11, metric 1
                      MultiRecv
```

Meaning The routing tables on Device R1 and Device R2 do not contain the default 0.0.0.0/0 route. This verifies that the default route is no longer present in the OSPF domain. To reactivate the ge-0/0/2.6 interface, issue the **activate interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30** configuration mode command.

Related Documentation

- *Understanding Conditionally Generated Routes*

Example: Using Routing Policy in an ISP Network

This example is a case study in how routing policies might be used in a typical Internet service provider (ISP) network.

- Requirements on page 150
- Overview on page 150
- Set Commands for All Devices in the Topology on page 151
- Configuring Device Customer-1 on page 157
- Configuring Device Customer-2 on page 159
- Configuring Devices ISP-1 and ISP-2 on page 163
- Configuring Device ISP-3 on page 168
- Configuring Device Exchange-2 on page 173
- Configuring Device Private-Peer-2 on page 175
- Verification on page 179

Requirements

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

Overview

In this network example, the ISP's AS number is 64510. The ISP has two transit peers (AS 64514 and AS 64515) to which it connects at an exchange point. The ISP is also connected to two private peers (AS 64513 and AS 64516) with which it exchanges specific customer routes. The ISP has two customers (AS 64511 and AS 64512).

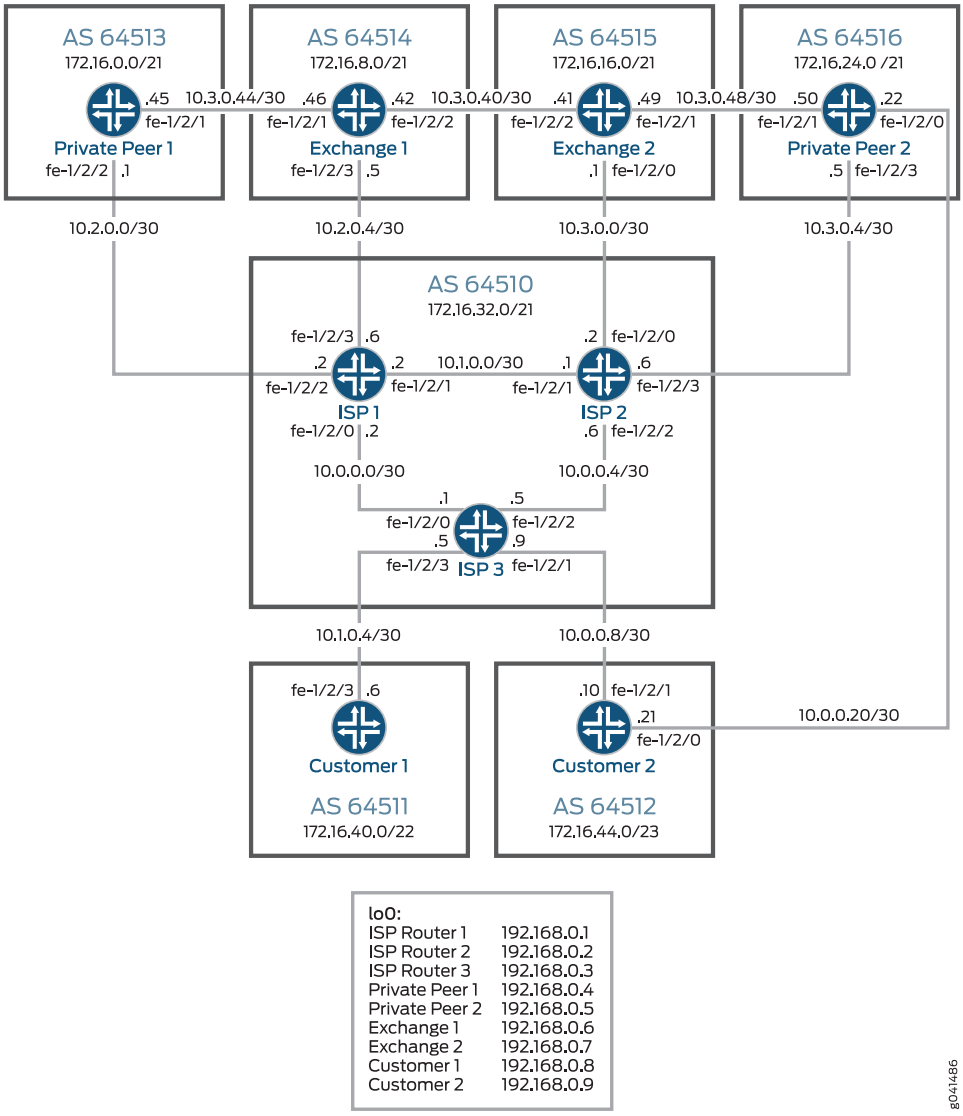
The ISP policies are configured in an outbound direction. That is, the example focuses on the routes that the ISP announces to its peers and customers, and includes the following:

1. The ISP has been assigned AS 64510 and the routing space of 172.16.32.0/21. With the exception of the two customer networks, all other customer routes are simulated with static routes.
2. The exchange peers are used for transit service to other portions of the Internet. This means that the ISP is accepting all routes (the full Internet routing table) from those BGP peers. To help maintain an optimized Internet routing table, the ISP is configured to advertise only two aggregate routes to the transit peers.
3. The ISP administrators want all data to the private peers to use the direct links. As a result, all the customer routes from the ISP are advertised to those private peers. These peers then advertise all their customer routes to the ISP.
4. Finally, each customer has a different set of requirements. Customer-1 requires a singe default route. Customer-2 requires specific routes.

Topology

Figure 19 on page 151 shows the sample network.

Figure 19: ISP Network Example



Set Commands for All Devices in the Topology

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 Customer-1

```
set interfaces fe-1/2/3 unit 0 description to_ISP-3
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.8/32
set protocols bgp group ext type external
set protocols bgp group ext export send-statics
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set policy-options policy-statement send-statics term static-routes from protocol static
```

```
set policy-options policy-statement send-statics term static-routes then accept
set routing-options static route 172.16.40.0/25 reject
set routing-options static route 172.16.40.128/25 reject
set routing-options static route 172.16.41.0/25 reject
set routing-options static route 172.16.41.128/25 reject
set routing-options autonomous-system 64511
```

Device Customer-2

```
set interfaces fe-1/2/1 unit 0 description to_ISP-3
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces fe-1/2/0 unit 0 description to-Private-Peer-2
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.21/30
set interfaces lo0 unit 0 family inet address 192.168.0.9/32
set protocols bgp group ext type external
set protocols bgp group ext import inbound-routes
set protocols bgp group ext export outbound-routes
set protocols bgp group ext neighbor 10.0.0.9 peer-as 64510
set protocols bgp group ext neighbor 10.0.0.22 peer-as 64516
set policy-options policy-statement inbound-routes term AS64510-primary from protocol
  bgp
set policy-options policy-statement inbound-routes term AS64510-primary from as-path
  AS64510-routes
set policy-options policy-statement inbound-routes term AS64510-primary then
  local-preference 200
set policy-options policy-statement inbound-routes term AS64510-primary then accept
set policy-options policy-statement inbound-routes term AS64516-backup from protocol
  bgp
set policy-options policy-statement inbound-routes term AS64516-backup from as-path
  AS64516-routes
set policy-options policy-statement inbound-routes term AS64516-backup then
  local-preference 50
set policy-options policy-statement inbound-routes term AS64516-backup then accept
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set policy-options policy-statement outbound-routes term internal-bgp-routes from
  protocol bgp
set policy-options policy-statement outbound-routes term internal-bgp-routes from
  as-path my-own-routes
set policy-options policy-statement outbound-routes term internal-bgp-routes then
  accept
set policy-options policy-statement outbound-routes term no-transit then reject
set policy-options as-path my-own-routes "()"
set policy-options as-path AS64510-routes "64510 .*"
set policy-options as-path AS64516-routes "64516 .*"
set routing-options static route 172.16.44.0/26 reject
set routing-options static route 172.16.44.64/26 reject
set routing-options static route 172.16.44.128/26 reject
set routing-options static route 172.16.44.192/26 reject
set routing-options autonomous-system 64512
```

Device ISP-1

```
set interfaces fe-1/2/0 unit 0 description to_ISP-3
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_ISP-2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_Private-Peer-1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.2/30
```

```
set interfaces fe-1/2/3 unit 0 description to_Exchange-1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64513 type external
set protocols bgp group to_64513 export private-peer
set protocols bgp group to_64513 peer-as 64513
set protocols bgp group to_64513 neighbor 10.2.0.1
set protocols bgp group to_64514 type external
set protocols bgp group to_64514 export exchange-peer
set protocols bgp group to_64514 peer-as 64514
set protocols bgp group to_64514 neighbor 10.2.0.5
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 protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  route-filter 172.16.32.0/21 exact
set policy-options policy-statement exchange-peer term AS64510-Aggregate then accept
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  route-filter 172.16.40.0/22 exact
set policy-options policy-statement exchange-peer term Customer-2-Aggregate then
  accept
set policy-options policy-statement exchange-peer term reject-all-other-routes then
  reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next-hop-self then next-hop self
set policy-options policy-statement private-peer term statics from protocol static
set policy-options policy-statement private-peer term statics then accept
set policy-options policy-statement private-peer term isp-and-customer-routes from
  protocol bgp
set policy-options policy-statement private-peer term isp-and-customer-routes from
  route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement private-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement private-peer term reject-all then reject
set routing-options static route 172.16.32.0/24 reject
set routing-options static route 172.16.33.0/24 reject
set routing-options aggregate route 172.16.32.0/21
set routing-options aggregate route 172.16.40.0/22
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device ISP-2

```
set interfaces fe-1/2/1 unit 0 description to_ISP-1
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces fe-1/2/2 unit 0 description to_ISP-3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.6/30
set interfaces fe-1/2/3 unit 0 description to_Private-Peer-2
```

```
set interfaces fe-1/2/3 unit 0 family inet address 10.3.0.6/30
set interfaces fe-1/2/0 unit 0 description to_Exchange-2
set interfaces fe-1/2/0 unit 0 family inet address 10.3.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group AS-64516 type external
set protocols bgp group AS-64516 export private-peer
set protocols bgp group AS-64516 peer-as 64516
set protocols bgp group AS-64516 neighbor 10.3.0.5
set protocols bgp group AS-64515 type external
set protocols bgp group AS-64515 export exchange-peer
set protocols bgp group AS-64515 peer-as 64515
set protocols bgp group AS-64515 neighbor 10.3.0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
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 exchange-peer term AS64510-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  route-filter 172.16.32.0/21 exact
set policy-options policy-statement exchange-peer term AS64510-Aggregate then accept
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  route-filter 172.16.44.0/23 exact
set policy-options policy-statement exchange-peer term Customer-2-Aggregate then
  accept
set policy-options policy-statement exchange-peer term reject-all-other-routes then
  reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next-hop-self then next-hop self
set policy-options policy-statement private-peer term statics from protocol static
set policy-options policy-statement private-peer term statics then accept
set policy-options policy-statement private-peer term isp-and-customer-routes from
  protocol bgp
set policy-options policy-statement private-peer term isp-and-customer-routes from
  route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement private-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement private-peer term reject-all then reject
set routing-options static route 172.16.34.0/24 reject
set routing-options static route 172.16.35.0/24 reject
set routing-options aggregate route 172.16.44.0/23
set routing-options aggregate route 172.16.32.0/21
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device ISP-3

```
set interfaces fe-1/2/0 unit 0 description to_ISP-1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 description to_ISP-2
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
```

```
set interfaces fe-1/2/3 unit 0 description to_Customer-1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/1 unit 0 description to_Customer-2
set interfaces fe-1/2/1 unit 0 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 int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 export customer-1-peer
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols bgp group to_64512 type external
set protocols bgp group to_64512 export customer-2-peer
set protocols bgp group to_64512 neighbor 10.0.0.10 peer-as 64512
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement customer-1-peer term defaut-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement customer-1-peer term defaut-route then accept
set policy-options policy-statement customer-1-peer term reject-all-other-routes then
  reject
set policy-options policy-statement customer-2-peer term statics from protocol static
set policy-options policy-statement customer-2-peer term statics then accept
set policy-options policy-statement customer-2-peer term isp-and-customer-routes
  from protocol bgp
set policy-options policy-statement customer-2-peer term isp-and-customer-routes
  from route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement customer-2-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement customer-2-peer term default-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement customer-2-peer term default-route then accept
set policy-options policy-statement customer-2-peer term reject-all-other-routes then
  reject
set policy-options policy-statement if-upstream-routes-exist term
  only-certain-contributing-routes from route-filter 172.16.8.0/21 exact
set policy-options policy-statement if-upstream-routes-exist term
  only-certain-contributing-routes then accept
set policy-options policy-statement if-upstream-routes-exist term reject-all-other-routes
  then reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next then next-hop self
set routing-options static route 172.16.36.0/24 reject
set routing-options static route 172.16.37.0/24 reject
set routing-options static route 172.16.38.0/24 reject
set routing-options static route 172.16.39.0/24 reject
set routing-options generate route 0.0.0.0/0 policy if-upstream-routes-exist
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device Exchange-1

```
set interfaces fe-1/2/3 unit 0 description to_ISP-1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.5/30
```

```
set interfaces fe-1/2/2 unit 0 description to_Exchange-2
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.42/30
set interfaces fe-1/2/1 unit 0 description to_Private-Peer-1
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.45/30
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols bgp group ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.2.0.6
set protocols bgp group ext neighbor 10.3.0.41 peer-as 64515
set policy-options policy-statement send-static from protocol static
set policy-options policy-statement send-static then accept
set routing-options static route 172.16.8.0/21 reject
set routing-options autonomous-system 64514
```

Device Exchange-2

```
set interfaces fe-1/2/0 unit 0 description to_ISP-2
set interfaces fe-1/2/0 unit 0 family inet address 10.3.0.1/30
set interfaces fe-1/2/2 unit 0 description to_Exchange-1
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.41/30
set interfaces fe-1/2/1 unit 0 description to_Private-Peer-2
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.49/30
set interfaces lo0 unit 0 family inet address 192.168.0.7/32
set protocols bgp group ext type external
set protocols bgp group ext export outbound-routes
set protocols bgp group ext neighbor 10.3.0.2 peer-as 64510
set protocols bgp group ext neighbor 10.3.0.50 peer-as 64516
set protocols bgp group ext neighbor 10.3.0.42 peer-as 64514
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set routing-options autonomous-system 64515
set routing-options static route 172.16.16.0/21 reject
```

Device Private-Peer-1

```
set interfaces fe-1/2/2 unit 0 description to_ISP-1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.1/30
set interfaces fe-1/2/1 unit 0 description to_Exchange-1
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.46/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.2.0.2
set routing-options autonomous-system 64513
```

Device Private-Peer-2

```
set interfaces fe-1/2/3 unit 0 description to_ISP-2
set interfaces fe-1/2/3 unit 0 family inet address 10.3.0.5/30
set interfaces fe-1/2/0 unit 0 description to_Customer-1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.22/30
set interfaces fe-1/2/1 unit 0 description to_Exchange-2
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.50/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols bgp group ext type external
set protocols bgp group ext export outbound-routes
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.3.0.6
set protocols bgp group to-64512 type external
set protocols bgp group to-64512 peer-as 64512
```

```
set protocols bgp group to-64512 neighbor 10.0.0.21
set protocols bgp group to-64512 export internal-routes
set protocols bgp group to-64515 type external
set protocols bgp group to-64515 export outbound-routes
set protocols bgp group to-64515 peer-as 64515
set protocols bgp group to-64515 neighbor 10.3.0.49
set policy-options policy-statement if-upstream-routes-exist term as-64515-routes from
  route-filter 172.16.16.0/21 exact
set policy-options policy-statement if-upstream-routes-exist term as-64515-routes then
  accept
set policy-options policy-statement if-upstream-routes-exist term reject-all-other-routes
  then reject
set policy-options policy-statement internal-routes term statics from protocol static
set policy-options policy-statement internal-routes term statics then accept
set policy-options policy-statement internal-routes term default-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement internal-routes term default-route then accept
set policy-options policy-statement internal-routes term reject-all-other-routes then
  reject
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set policy-options policy-statement outbound-routes term allowed-bgp-routes from
  as-path my-own-routes
set policy-options policy-statement outbound-routes term allowed-bgp-routes from
  as-path AS64512-routes
set policy-options policy-statement outbound-routes term allowed-bgp-routes then
  accept
set policy-options policy-statement outbound-routes term no-transit then reject
set policy-options as-path my-own-routes "()"
set policy-options as-path AS64512-routes 64512
set routing-options static route 172.16.24.0/25 reject
set routing-options static route 172.16.24.128/25 reject
set routing-options static route 172.16.25.0/26 reject
set routing-options static route 172.16.25.64/26 reject
set routing-options generate route 0.0.0.0/0 policy if-upstream-routes-exist
set routing-options autonomous-system 64516
```

Configuring Device Customer-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 *CLI User Guide*.

Device Customer-1 has multiple static routes configured to simulate customer routes. These routes are sent to the ISP.

To configure Device Customer-1:

1. Configure the device interfaces.

```
[edit interfaces]
user@Customer-1# set fe-1/2/3 unit 0 description to_ISP-3
user@Customer-1# set fe-1/2/3 unit 0 family inet address 10.1.0.6/30

user@Customer-1# set lo0 unit 0 family inet address 192.168.0.8/32
```

2. Configure the static routes.

```
[edit routing-options static]
user@Customer-1# set route 172.16.40.0/25 reject
user@Customer-1# set route 172.16.40.128/25 reject
user@Customer-1# set route 172.16.41.0/25 reject
user@Customer-1# set route 172.16.41.128/25 reject
```

3. Configure the policy to send static routes.

```
[edit policy-options policy-statement send-statics term static-routes]
user@Customer-1# set from protocol static
user@Customer-1# set then accept
```

4. Configure the external BGP (EBGP) connection to the ISP.

```
[edit protocols bgp group ext]
user@Customer-1# set type external
user@Customer-1# set export send-statics
user@Customer-1# set peer-as 64510
user@Customer-1# set neighbor 10.1.0.5
```

5. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Customer-1# 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.

```
user@Customer-1# show interfaces
fe-1/2/1 {
    unit 0 {
        description to_ISP-3;
        family inet {
            address 10.1.0.6/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.8/32;
        }
    }
}
```

```
user@Customer-1# show protocols
bgp {
    group ext {
        type external;
        export send-statics;
        peer-as 64510;
        neighbor 10.1.0.5;
    }
}
```

```
user@Customer-1# show policy-options
policy-statement send-statics {
   term static-routes {
      from protocol static;
      then accept;
   }
}

user@Customer-1# show routing-options
static {
   route 172.16.40.0/25 reject;
   route 172.16.40.128/25 reject;
   route 172.16.41.0/25 reject;
   route 172.16.41.128/25 reject;
}
autonomous-system 64511;
```

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

Configuring Device Customer-2

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*.

Device Customer-2 has two static routes configured to simulate customer routes. These routes are sent to the ISP. Customer-2 has a link to the ISP, as well as a link to AS 8000. This customer has requested specific customer routes from the ISP, as well as from AS 64516. Customer-2 wants to use the ISP for transit service to the Internet, and has requested a default route from the ISP.

To configure Device Customer-2:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@Customer-2# set fe-1/2/1 unit 0 description to_ISP-3
   user@Customer-2# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30

   user@Customer-2# set fe-1/2/0 unit 0 description to-Private-Peer-2
   user@Customer-2# set fe-1/2/0 unit 0 family inet address 10.0.0.21/30

   user@Customer-2# set lo0 unit 0 family inet address 192.168.0.9/32
   ```

2. Configure the static routes.

   ```
   [edit routing-options static]
   user@Customer-2# set route 172.16.44.0/26 reject
   user@Customer-2# set route 172.16.44.64/26 reject
   user@Customer-2# set route 172.16.44.128/26 reject
   user@Customer-2# set route 172.16.44.192/26 reject
   ```

3. Configure the import routing policy.

   The route with the highest local preference value is preferred. Routes from the ISP are preferred over the same routes from Device Private-Peer-2

```
[edit policy-options policy-statement inbound-routes]
user@Customer-2# set term AS64510-primary from protocol bgp
user@Customer-2# set term AS64510-primary from as-path AS64510-routes
user@Customer-2# set term AS64510-primary then local-preference 200
user@Customer-2# set term AS64510-primary then accept

[edit policy-options policy-statement inbound-routes]
user@Customer-2# set term AS64516-backup from protocol bgp
user@Customer-2# set term AS64516-backup from as-path AS64516-routes
user@Customer-2# set term AS64516-backup then local-preference 50
user@Customer-2# set term AS64516-backup then accept

[edit policy-options]
user@Customer-2# set as-path AS64510-routes "64510 .*"
user@Customer-2# set as-path AS64516-routes "64516 .*"
```

4. Configure the export routing policy.

```
[edit policy-options policy-statement outbound-routes]
user@Customer-2# set term statics from protocol static
user@Customer-2# set term statics then accept

user@Customer-2# set term internal-bgp-routes from protocol bgp
user@Customer-2# set term internal-bgp-routes from as-path my-own-routes
user@Customer-2# set term internal-bgp-routes then accept
user@Customer-2# set term no-transit then reject

[edit policy-options]
user@Customer-2# set as-path my-own-routes "()"
```

5. Configure the external BGP (EBGP) connection to the ISP and to Device Private-Peer-2.

```
[edit protocols bgp group ext]
user@Customer-2# set type external
user@Customer-2# set import inbound-routes
user@Customer-2# set export outbound-routes
user@Customer-2# set neighbor 10.0.0.9 peer-as 64510
user@Customer-2# set neighbor 10.0.0.22 peer-as 64516
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Customer-2# set autonomous-system 64512
```

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@Customer-2# show interfaces
fe-1/2/1 {
  unit 0 {
    description to_ISP-3;
    family inet {
```



```
        address 10.0.0.10/30;
      }
    }
}
fe-1/2/0 {
    unit 0 {
      description to-Private-Peer-2;
      family inet {
        address 10.0.0.21/30;
      }
    }
}
lo0 {
    unit 0 {
      family inet {
        address 192.168.0.9/32;
      }
    }
}
```

```
user@Customer-2# show protocols
bgp {
    group ext {
      type external;
      import inbound-routes;
      export outbound-routes;
      neighbor 10.0.0.9 {
        peer-as 64510;
      }
      neighbor 10.0.0.22 {
        peer-as 64516;
      }
    }
}
```

```
user@Customer-2# show policy-options
policy-statement inbound-routes {
    term AS64510-primary {
      from {
        protocol bgp;
        as-path AS64510-routes;
      }
      then {
        local-preference 200;
        accept;
      }
    }
    term AS64516-backup {
      from {
        protocol bgp;
        as-path AS64516-routes;
      }
      then {
        local-preference 50;
        accept;
      }
    }
```

```
}
policy-statement outbound-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term internal-bgp-routes {
        from {
            protocol bgp;
            as-path my-own-routes;
        }
        then accept;
    }
    term no-transit {
        then reject;
    }
}
as-path my-own-routes "()";
as-path AS64510-routes "64510 .*";
as-path AS64516-routes "64516 .*";
```

```
user@Customer-2# show routing-options
static {
    route 172.16.44.0/26 reject;
    route 172.16.44.64/26 reject;
    route 172.16.44.128/26 reject;
    route 172.16.44.192/26 reject;
}
autonomous-system 64512;
```

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

Configuring Devices ISP-1 and ISP-2

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*.

Device ISP-1 and Device ISP-2 each have two policies configured: The **private-peer** policy and the **exchange-peer** policy. Because of their similar configurations, this example shows the step-by-step configuration only for Device ISP-2.

On Device ISP-2, the private-peer policy sends the ISP customer routes to Device Private-Peer-2. The policy accepts all local static routes (local Device ISP-2 customers) and all BGP routes in the 172.16.32.0/21 range (advertised by other ISP routers). These two policy terms represent the ISP customer routes. The final policy term rejects all other routes, which includes the entire Internet routing table sent by the exchange peers. These routes do not need to be sent to Device Private-Peer-2 for two reasons:

- The peer already maintains a connection to Device Exchange-2 in our example, so the routes are redundant.
- The private peer wants customer routes only. The **private-peer** policy accomplishes this goal. The **exchange-peer** policy sends routes to Device Exchange-2.

In the example, only two routes need to be sent to Device Exchange-2:

- The aggregate route that represents the AS 64510 routing space of 172.16.32.0/21. This route is configured as an aggregate route locally and is advertised by the **exchange-peer** policy.
- The address space assigned to Customer-2, 172.16.44.0/23. This smaller aggregate route needs to be sent to Device Exchange-2 because the customer is also attached to the AS 64516 peer (Device Private-Peer-2).

Sending these two routes to Device Exchange-2 allows other networks in the Internet to reach the customer through either the ISP or the private peer. If just the private peer were to advertise the /23 network while the ISP maintained only its /21 aggregate, all traffic destined for the customer would transit AS 64516 only. Because the customer also wants routes from the ISP, the 172.16.44.0/23 route is announced by Device ISP-2. Like the larger aggregate route, the 172.16.44.0/23 route is configured locally and is advertised by the exchange-peer policy. The final term in that policy rejects all routes, including the specific customer networks of the ISP, the customer routes from Device Private-Peer-1, the customer routes from Device Private-Peer-2, and the routing table from Device Exchange-1. In essence, this final term prevents the ISP from performing transit services for the Internet at large.

To configure Device ISP-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@ISP-2# set fe-1/2/1 unit 0 description to_ISP-1
user@ISP-2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30
```

```
user@ISP-2# set fe-1/2/2 unit 0 description to_ISP-3
user@ISP-2# set fe-1/2/2 unit 0 family inet address 10.0.0.6/30

user@ISP-2# set fe-1/2/3 unit 0 description to_Private-Peer-2
user@ISP-2# set fe-1/2/3 unit 0 family inet address 10.3.0.6/30

user@ISP-2# set fe-1/2/0 unit 0 description to_Exchange-2
user@ISP-2# set fe-1/2/0 unit 0 family inet address 10.3.0.2/30

user@ISP-2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@ISP-2# set interface fe-1/2/2.0
user@ISP-2# set interface fe-1/2/1.0
user@ISP-2# set interface lo0.0 passive
```

3. Configure the static and aggregate routes.

```
[edit routing-options static]
user@ISP-2# set route 172.16.34.0/24 reject
user@ISP-2# set route 172.16.35.0/24 reject

[edit routing-options aggregate]
user@ISP-2# set route 172.16.44.0/23
user@ISP-2# set route 172.16.32.0/21
```

4. Configure the routing policies for the exchange peers.

```
[edit policy-options policy-statement exchange-peer]
user@ISP-2# set term AS64510-Aggregate from protocol aggregate
user@ISP-2# set term AS64510-Aggregate from route-filter 172.16.32.0/21 exact
user@ISP-2# set term AS64510-Aggregate then accept
user@ISP-2# set term Customer-2-Aggregate from protocol aggregate
user@ISP-2# set term Customer-2-Aggregate from route-filter 172.16.44.0/23 exact
user@ISP-2# set term Customer-2-Aggregate then accept
user@ISP-2# set term reject-all-other-routes then reject
```

5. Configure the routing policies for the internal peers.

```
[edit policy-options policy-statement internal-peers]
user@ISP-2# set term statics from protocol static
user@ISP-2# set term statics then accept
user@ISP-2# set term next-hop-self then next-hop self
```

6. Configure the routing policies for the private peer.

```
[edit policy-options policy-statement private-peer]
user@ISP-2# set term statics from protocol static
user@ISP-2# set term statics then accept
user@ISP-2# set term isp-and-customer-routes from protocol bgp
user@ISP-2# set term isp-and-customer-routes from route-filter 172.16.32.0/21
   orlonger
user@ISP-2# set term isp-and-customer-routes then accept
user@ISP-2# set term reject-all then reject
```

7. Configure the internal BGP (IBGP) connections to the other ISP devices.

```
[edit protocols bgp group int]
user@ISP-2# set type internal
user@ISP-2# set local-address 192.168.0.2
user@ISP-2# set export internal-peers
user@ISP-2# set neighbor 192.168.0.1
user@ISP-2# set neighbor 192.168.0.3
```

8. Configure the EBGP connections to the exchange peer and the private peer.

```
[edit protocols bgp group AS-64516]
user@ISP-2# set type external
user@ISP-2# set export private-peer
user@ISP-2# set peer-as 64516
user@ISP-2# set neighbor 10.3.0.5

[edit protocols bgp group AS-64515]
user@ISP-2# set type external
user@ISP-2# set export exchange-peer
user@ISP-2# set peer-as 64515
user@ISP-2# set neighbor 10.3.0.1
```

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

```
[edit routing-options]
user@ISP-2# set router-id 192.168.0.2
user@ISP-2# set autonomous-system 64510
```

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@ISP-2# show interfaces
fe-1/2/0 {
  unit 0{
    description to_Exchange-2;
    family inet {
      address 10.3.0.2/30;
    }
  }
}
fe-1/2/1 {
  unit 0{
    description to_ISP-1;
    family inet {
      address 10.1.0.1/30;
    }
  }
}
fe-1/2/2 {
  unit 0 {
    description to_ISP-3;
    family inet {
      address 10.0.0.6/30;
    }
  }
```

```
}
fe-1/2/3 {
    unit 0 {
        description to_Private-Peer-2;
        family inet {
            address 10.3.0.6/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@ISP-2# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.2;
        export internal-peers;
        neighbor 192.168.0.1;
        neighbor 192.168.0.3;
    }
    group AS-64516 {
        type external;
        export private-peer;
        peer-as 64516;
        neighbor 10.3.0.5;
    }
    group AS-64515 {
        type external;
        export exchange-peer;
        peer-as 64515;
        neighbor 10.3.0.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/2.0;
        interface fe-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@ISP-2# show policy-options
policy-statement exchange-peer {
    term AS64510-Aggregate {
        from {
            protocol aggregate;
            route-filter 172.16.32.0/21 exact;
        }
        then accept;
```

```
    }
    term Customer-2-Aggregate {
      from {
        protocol aggregate;
        route-filter 172.16.44.0/23 exact;
      }
      then accept;
    }
    term reject-all-other-routes {
      then reject;
    }
  }
  policy-statement internal-peers {
    term statics {
      from protocol static;
      then accept;
    }
    term next-hop-self {
      then {
        next-hop self;
      }
    }
  }
  policy-statement private-peer {
    term statics {
      from protocol static;
      then accept;
    }
    term isp-and-customer-routes {
      from {
        protocol bgp;
        route-filter 172.16.32.0/21 orlonger;
      }
      then accept;
    }
    term reject-all {
      then reject;
    }
  }
```

```
user@ISP-2# show routing-options
static {
  route 172.16.34.0/24 reject;
  route 172.16.35.0/24 reject;
}
aggregate {
  route 172.16.44.0/23;
  route 172.16.32.0/21;
}
router-id 192.168.0.2;
autonomous-system 64510;
```

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

Configuring Device ISP-3

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*.

On Device ISP-3, a separate policy is in place for each customer. The default route for Customer-1 is being sent by the **customer-1-peer** policy. This policy finds the 0.0.0.0/0 default route in inet.0 and accepts it. The policy also rejects all other routes, thereby not sending all BGP routes on the ISP router. The **customer-2-peer** policy is for Customer-2 and contains the same policy terms, which also send the default route and no other transit BGP routes. The additional terms in the **customer-2-peer** policy send the ISP customer routes to Customer-2. Because there are local static routes on Device ISP-3 that represent local customers, these routes are sent as well as all other internal routes announced to the local router by the other ISP routers.

If the upstream route from Device Exchange-1 (172.16.8.0/21) is present, Device ISP-3 generates a default route.

To configure Device ISP-3:

1. Configure the device interfaces.

```
[edit interfaces]
user@ISP-3# set fe-1/2/0 unit 0 description to_ISP-1
user@ISP-3# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@ISP-3# set fe-1/2/2 unit 0 description to_ISP-2
user@ISP-3# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@ISP-3# set fe-1/2/3 unit 0 description to_Customer-1
user@ISP-3# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@ISP-3# set fe-1/2/1 unit 0 description to_Customer-2
user@ISP-3# set fe-1/2/1 unit 0 family inet address 10.0.0.9/30

user@ISP-3# set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure the interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@ISP-3# set interface fe-1/2/0.0
user@ISP-3# set interface fe-1/2/2.0
user@ISP-3# set interface lo0.0 passive
```

3. Configure the static routes.

```
[edit routing-options static]
user@ISP-3# set route 172.16.36.0/24 reject
user@ISP-3# set route 172.16.37.0/24 reject
user@ISP-3# set route 172.16.38.0/24 reject
user@ISP-3# set route 172.16.39.0/24 reject
```

4. Configure a routing policy that generates a default static route only if a certain upstream route exists.

```
[edit policy-options policy-statement if-upstream-routes-exist term
  only-certain-contributing-routes]
user@ISP-3# set from route-filter 172.16.8.0/21 exact
user@ISP-3# set then accept

[edit policy-options policy-statement if-upstream-routes-exist]
user@ISP-3# set term reject-all-other-routes then reject

[edit routing-options generate route 0.0.0.0/0]
user@ISP-3# set policy if-upstream-routes-exist
```

5. Configure the routing policy for Customer-1.

```
[edit policy-options policy-statement customer-1-peer]
user@ISP-3# set term defaut-route from route-filter 0.0.0.0/0 exact
user@ISP-3# set term defaut-route then accept
user@ISP-3# set term reject-all-other-routes then reject
```

6. Configure the routing policy for Customer-2.

```
[edit policy-options policy-statement customer-2-peer]
user@ISP-3# set term statics from protocol static
user@ISP-3# set term statics then accept
user@ISP-3# set term isp-and-customer-routes from protocol bgp
user@ISP-3# set term isp-and-customer-routes from route-filter 172.16.32.0/21
  orlonger
user@ISP-3# set term isp-and-customer-routes then accept
user@ISP-3# set term default-route from route-filter 0.0.0.0/0 exact
user@ISP-3# set term default-route then accept
user@ISP-3# set term reject-all-other-routes then reject
```

7. Configure the routing policies for the internal peers.

```
[edit policy-options policy-statement internal-peers]
user@ISP-3# set term statics from protocol static
user@ISP-3# set term statics then accept
user@ISP-3# set term next then next-hop self
```

8. Configure the internal BGP (IBGP) connections to the other ISP devices.

```
[edit protocols bgp group int]
user@ISP-3# set type internal
user@ISP-3# set local-address 192.168.0.3
user@ISP-3# set export internal-peers
user@ISP-3# set neighbor 192.168.0.1
user@ISP-3# set neighbor 192.168.0.2
```

9. Configure the EBGP connections to the customer peers.

```
[edit protocols bgp group to_64511]
user@ISP-3# set type external
user@ISP-3# set export customer-1-peer
user@ISP-3# set neighbor 10.1.0.6 peer-as 64511

[edit protocols bgp group to_64512]
```

```
user@ISP-3# set type external
user@ISP-3# set export customer-2-peer
user@ISP-3# set neighbor 10.0.0.10 peer-as 64512
```

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

```
[edit routing-options]
user@ISP-3# set router-id 192.168.0.3
user@ISP-3# set autonomous-system 64510
```

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@ISP-3# show interfaces
fe-1/2/0 {
    unit 0 {
        description to_ISP-1;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        description to_Customer-2;
        family inet {
            address 10.0.0.9/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        description to_ISP-2;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        description to_Customer-1;
        family inet {
            address 10.1.0.5/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.3/32;
        }
    }
}
```

```
user@ISP-3# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.3;
        export internal-peers;
        neighbor 192.168.0.1;
        neighbor 192.168.0.2;
    }
    group to_64511 {
        type external;
        export customer-1-peer;
        neighbor 10.1.0.6 {
            peer-as 64511;
        }
    }
    group to_64512 {
        type external;
        export customer-2-peer;
        neighbor 10.0.0.10 {
            peer-as 64512;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.0;
        interface fe-1/2/2.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@ISP-3# show policy-options
policy-statement customer-1-peer {
    term defaut-route {
        from {
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term reject-all-other-routes {
        then reject;
    }
}
policy-statement customer-2-peer {
    term statics {
        from protocol static;
        then accept;
    }
    term isp-and-customer-routes {
        from {
            protocol bgp;
            route-filter 172.16.32.0/21 orlonger;
        }
```

```
        then accept;
      }
      term default-route {
        from {
          route-filter 0.0.0.0/0 exact;
        }
        then accept;
      }
      term reject-all-other-routes {
        then reject;
      }
    }
    policy-statement if-upstream-routes-exist {
      term only-certain-contributing-routes {
        from {
          route-filter 172.16.8.0/21 exact;
        }
        then accept;
      }
      term reject-all-other-routes {
        then reject;
      }
    }
    policy-statement internal-peers {
      term statics {
        from protocol static;
        then accept;
      }
      term next {
        then {
          next-hop self;
        }
      }
    }

    user@ISP-3# show routing-options
    static {
      route 172.16.36.0/24 reject;
      route 172.16.37.0/24 reject;
      route 172.16.38.0/24 reject;
      route 172.16.39.0/24 reject;
    }
    generate {
      route 0.0.0.0/0 policy if-upstream-routes-exist;
    }
    router-id 192.168.0.3;
    autonomous-system 64510;
```

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

Configuring Device Exchange-2

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*.

Device Exchange-2 exchanges all BGP routes with all BGP peers. The outbound-routes policy for Device Exchange-2 advertises locally defined static routes using BGP. The exclusion of a final **then reject** term causes the default BGP export policy to take effect, which is to send all BGP routes to all external BGP peers.

To configure Device Exchange-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@Exchange-2# set fe-1/2/0 unit 0 description to_ISP-2
user@Exchange-2# set fe-1/2/0 unit 0 family inet address 10.3.0.1/30

user@Exchange-2# set fe-1/2/2 unit 0 description to_Exchange-1
user@Exchange-2# set fe-1/2/2 unit 0 family inet address 10.3.0.41/30

user@Exchange-2# set fe-1/2/1 unit 0 description to_Private-Peer-2
user@Exchange-2# set fe-1/2/1 unit 0 family inet address 10.3.0.49/30

user@Exchange-2# set lo0 unit 0 family inet address 192.168.0.7/32
```

2. Configure the static routes.

```
[edit routing-options static]
set route 172.16.16.0/21 reject
```

3. Configure a routing policy that generates a default static route only if certain internal routes exist.

```
[edit policy-options policy-statement outbound-routes term statics]
user@Exchange-2# set from protocol static
user@Exchange-2# set then accept
```

4. Configure the EBGP connections to the customer peers.

```
[edit protocols bgp group ext]
user@Exchange-2# set type external
user@Exchange-2# set export outbound-routes
user@Exchange-2# set neighbor 10.3.0.2 peer-as 64510
user@Exchange-2# set neighbor 10.3.0.50 peer-as 64516
user@Exchange-2# set neighbor 10.3.0.42 peer-as 64514
```

5. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Exchange-2# set autonomous-system 64515
```

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@Exchange-2 show interfaces
fe-1/2/0 {
    unit 0 {
        description to_ISP-2;
        family inet {
            address 10.3.0.1/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        description to_Private-Peer-2;
        family inet {
            address 10.3.0.49/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        description to_Exchange-1;
        family inet {
            address 10.3.0.41/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.7/32;
        }
    }
}
```

```
user@Exchange-2# show protocols
bgp {
    group ext {
        type external;
        export outbound-routes;
        neighbor 10.3.0.2 {
            peer-as 64510;
        }
        neighbor 10.3.0.50 {
            peer-as 64516;
        }
        neighbor 10.3.0.42 {
            peer-as 64514;
        }
    }
}
```

```
user@Exchange-2# show policy-options
policy-statement outbound-routes {
    term statics {
```

```
        from protocol static;
        then accept;
    }
}
```

```
user@Exchange-2# show routing-options
static {
    route 172.16.16.0/21 reject;
}
autonomous-system 64515;
```

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

Configuring Device Private-Peer-2

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*.

Device Private-Peer-2 performs two main functions:

- Advertises routes local to AS 64516 to both the exchange peers and the ISP routers. The **outbound-routes** policy advertises the local static routes (that is, customers) on the router, and also advertises all routes learned by BGP that originated in either AS 64516 or AS 64512. These routes include other AS 64516 customer routes in addition to the AS 64512 customer. The AS routes are identified by an AS path regular expression match criteria in the policy.
- Advertises the 0.0.0.0/0 default route to the AS 64512 customer router. To accomplish this, the private peer creates a generated route for 0.0.0.0/0 locally on the router. This generated route is further assigned a policy called **if-upstream-routes-exist**, which allows only certain routes to contribute to the generated route, making it an active route in the routing table. Once the route is active, it can be sent to the AS 64512 router using BGP and the configured policies. The **if-upstream-routes-exist** policy accepts only the 172.16.32.0/21 route from Device Exchange-2, and rejects all other routes. If the 172.16.32.0/21 route is withdrawn by the exchange peer, the private peer loses the 0.0.0.0/0 default route and withdraws the default route from the AS 64512 customer router.

To configure Device Private-Peer-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@Private-Peer-2# set fe-1/2/3 unit 0 description to_ISP-2
user@Private-Peer-2# set fe-1/2/3 unit 0 family inet address 10.3.0.5/30

user@Private-Peer-2# set fe-1/2/0 unit 0 description to_Customer-1
user@Private-Peer-2# set fe-1/2/0 unit 0 family inet address 10.0.0.22/30

user@Private-Peer-2# set fe-1/2/1 unit 0 description to_Exchange-2
user@Private-Peer-2# set fe-1/2/1 unit 0 family inet address 10.3.0.50/30
```

```
user@Private-Peer-2# set lo0 unit 0 family inet address 192.168.0.5/32
```

2. Configure the static routes.

```
[edit routing-options static]
user@Private-Peer-2# set route 172.16.24.0/25 reject
user@Private-Peer-2# set route 172.16.24.128/25 reject
user@Private-Peer-2# set route 172.16.25.0/26 reject
user@Private-Peer-2# set route 172.16.25.64/26 reject
```

3. Configure a routing policy that generates a default static route only if certain internal routes exist.

```
[edit policy-options policy-statement if-upstream-routes-exist]
user@Private-Peer-2# set term as-64515-routes from route-filter 172.16.16.0/21
  exact
user@Private-Peer-2# set term as-64515-routes then accept
user@Private-Peer-2# set term reject-all-other-routes then reject

[edit routing-options generate route 0.0.0.0/0]
user@Private-Peer-2# set policy if-upstream-routes-exist
```

4. Configure the routing policy that advertises local static routes and the default route.

```
[edit policy-options policy-statement internal-routes]
user@Private-Peer-2# set term statics from protocol static
user@Private-Peer-2# set term statics then accept
user@Private-Peer-2# set term default-route from route-filter 0.0.0.0/0 exact
user@Private-Peer-2# set term default-route then accept
user@Private-Peer-2# set term reject-all-other-routes then reject
```

5. Configure the routing policy that advertises local customer routes.

```
[edit policy-options policy-statement outbound-routes]
user@Private-Peer-2# set term statics from protocol static
user@Private-Peer-2# set term statics then accept
user@Private-Peer-2# set term allowed-bgp-routes from as-path my-own-routes
user@Private-Peer-2# set term allowed-bgp-routes from as-path AS64512-routes
user@Private-Peer-2# set term allowed-bgp-routes then accept
user@Private-Peer-2# set term no-transit then reject

[edit policy-options]
user@Private-Peer-2# set as-path my-own-routes "()"
user@Private-Peer-2# set as-path AS64512-routes 64512
```

6. Configure the EBGP connection to Customer-2.

```
[edit protocols bgp group to-64512]
user@Private-Peer-2# set type external
user@Private-Peer-2# set export internal-routes
user@Private-Peer-2# set peer-as 64512
user@Private-Peer-2# set neighbor 10.0.0.21
```

7. Configure the EBGP connection to Device Exchange-2.

```
[edit protocols bgp group to-64515]
user@Private-Peer-2# set type external
user@Private-Peer-2# set export outbound-routes
user@Private-Peer-2# set peer-as 64515
```

```
user@Private-Peer-2# set neighbor 10.3.0.49
```

8. Configure the EBGP connections to the ISP.

```
[edit protocols bgp group ext]
user@Private-Peer-2# set type external
user@Private-Peer-2# set export outbound-routes
user@Private-Peer-2# set peer-as 64510
user@Private-Peer-2# set neighbor 10.3.0.6
```

9. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Private-Peer-2# set autonomous-system 64516
```

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@Private-Peer-2# show interfaces
fe-1/2/0 {
  unit 0 {
    description to_Customer-1;
    family inet {
      address 10.0.0.22/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    description to_Exchange-2;
    family inet {
      address 10.3.0.50/30;
    }
  }
}
fe-1/2/3 {
  unit 0 {
    description to_ISP-2;
    family inet {
      address 10.3.0.5/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.5/32;
    }
  }
}

user@Private-Peer-2# show protocols
bgp {
  group ext {
    type external;
```

```
      export outbound-routes;
      peer-as 64510;
      neighbor 10.3.0.6;
    }
    group to-64512 {
      type external;
      export internal-routes;
      peer-as 64512;
      neighbor 10.0.0.21;
    }
    group to-64515 {
      type external;
      export outbound-routes;
      peer-as 64515;
      neighbor 10.3.0.49;
    }
}
```

```
user@Private-Peer-2# show policy-options
policy-statement if-upstream-routes-exist {
    term as-64515-routes {
        from {
            route-filter 172.16.16.0/21 exact;
        }
        then accept;
    }
    term reject-all-other-routes {
        then reject;
    }
}
policy-statement internal-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term default-route {
        from {
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term reject-all-other-routes {
        then reject;
    }
}
policy-statement outbound-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term allowed-bgp-routes {
        from as-path [ my-own-routes AS64512-routes ];
        then accept;
    }
    term no-transit {
        then reject;
```

```
    }
}
as-path my-own-routes "()";
as-path AS64512-routes 64512;

user@Private-Peer-2# show routing-options
static {
    route 172.16.24.0/25 reject;
    route 172.16.24.128/25 reject;
    route 172.16.25.0/26 reject;
    route 172.16.25.64/26 reject;
}
generate {
    route 0.0.0.0/0 policy if-upstream-routes-exist;
}
autonomous-system 64516;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device Customer-1 on page 179
- Verifying the Routes on Device Customer-2 on page 180
- Verifying the Routes on Device ISP-1 on page 182
- Verifying the Routes on Device ISP-2 on page 185
- Verifying the Routes on Device ISP-3 on page 188
- Verifying the Routes on Device Exchange-1 on page 190
- Verifying the Routes on Device Exchange-2 on page 192
- Verifying the Routes on Device Private-Peer-1 on page 194
- Verifying the Routes on Device Private-Peer-2 on page 195

Verifying the Routes on Device Customer-1

Purpose On Device Customer-1, check the routes in the routing table.

Action

```
user@Customer-1> show route

inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:09:25, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.1.0.5 via fe-1/2/3.0
10.1.0.4/30        *[Direct/0] 23:50:20
                    > via fe-1/2/3.0
10.1.0.6/32        *[Local/0] 5d 21:56:47
                      Local via fe-1/2/3.0
172.16.40.0/25     *[Static/5] 22:59:04
                      Reject
172.16.40.128/25   *[Static/5] 22:59:04
                      Reject
172.16.41.0/25     *[Static/5] 22:59:04
                      Reject
172.16.41.128/25   *[Static/5] 22:59:04
                      Reject
192.168.0.8/32     *[Direct/0] 5d 21:25:45
                    > via lo0.0
```

Meaning Device Customer-1 has its four static routes, and it has learned the default route through BGP.

Verifying the Routes on Device Customer-2

Purpose On Device Customer-2, check the routes in the routing table.

Action

```
user@Customer-2> show route
inet.0: 22 destinations, 23 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:10:35, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
                    [BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
10.0.0.8/30        *[Direct/0] 23:51:29
                    > via fe-1/2/0.10
10.0.0.10/32       *[Local/0] 23:52:49
                      Local via fe-1/2/0.10
10.0.0.20/30       *[Direct/0] 23:52:49
                    > via fe-1/2/0.0
10.0.0.21/32       *[Local/0] 23:52:49
                      Local via fe-1/2/0.0
172.16.24.0/25     *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.24.128/25   *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.25.0/26     *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.25.64/26    *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.32.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.33.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.34.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.35.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.36.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.37.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.38.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.39.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.44.0/26     *[Static/5] 22:57:28
                      Reject
172.16.44.64/26    *[Static/5] 22:57:28
                      Reject
172.16.44.128/26   *[Static/5] 22:57:28
                      Reject
```

```
172.16.44.192/26   *[Static/5] 22:57:28
                      Reject
192.168.0.9/32     *[Direct/0] 23:52:49
                    > via lo0.0
```

Meaning Device Customer-2 has learned the default route through its session with the ISP and also through its session with the private peer. The route learned from the ISP is preferred because it has a higher local preference.

Verifying the Routes on Device ISP-1

Purpose On Device ISP-1, check the routes in the routing table.

Action

```
user@ISP-1> show route
inet.0: 42 destinations, 53 routes (42 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 22:44:26, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
10.0.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/0.0
10.0.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/0.0
10.0.0.4/30        *[OSPF/10] 23:51:06, metric 2
                      to 10.1.0.1 via fe-1/2/1.0
                    > to 10.0.0.1 via fe-1/2/0.0
10.0.0.20/30       *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                     [BGP/170] 23:51:28, localpref 100
                       AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
10.1.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/1.0
10.1.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/1.0
10.2.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/2.0
10.2.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/2.0
10.2.0.4/30        *[Direct/0] 23:52:00
                    > via fe-1/2/3.0
10.2.0.6/32        *[Local/0] 23:52:00
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:51:28, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
10.3.0.48/30       *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:11:08, localpref 100
                      AS path: 64514 I, validation-state: unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.16.0/21     *[BGP/170] 02:02:10, localpref 100, from 192.168.0.2
                      AS path: 64515 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                     [BGP/170] 02:02:10, localpref 100
                       AS path: 64514 64515 I, validation-state: unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.24.0/25     *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                     [BGP/170] 23:06:33, localpref 100
                       AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.24.128/25   *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                     [BGP/170] 23:06:33, localpref 100
```

```
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.25.0/26     *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:06:33, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.25.64/26    *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:06:33, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.32.0/21     *[Aggregate/130] 22:44:27
                      Reject
172.16.32.0/24     *[Static/5] 22:44:27
                      Reject
172.16.33.0/24     *[Static/5] 22:44:27
                      Reject
172.16.34.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
172.16.35.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
172.16.36.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.37.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.38.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.39.0/24     *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.40.0/22     *[Aggregate/130] 22:44:27
                      Reject
172.16.40.0/25     *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.40.128/25   *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.41.0/25     *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.41.128/25   *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
172.16.44.0/26     *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
```

```
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.64/26     *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.128/26    *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.192/26    *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
192.168.0.1/32      *[Direct/0] 23:52:01
                    > via lo0.0
192.168.0.2/32      *[OSPF/10] 23:51:06, metric 1
                    > to 10.1.0.1 via fe-1/2/1.0
192.168.0.3/32      *[OSPF/10] 23:51:06, metric 1
                    > to 10.0.0.1 via fe-1/2/0.0
192.168.0.5/32      *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:51:28, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
224.0.0.5/32        *[OSPF/10] 23:52:07, metric 1
                      MultiRecv
```

Verifying the Routes on Device ISP-2

Purpose On Device ISP-2, check the routes in the routing table.

Action

```
user@ISP-2> show route
inet.0: 41 destinations, 59 routes (41 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 22:45:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
10.0.0.0/30        *[OSPF/10] 23:52:25, metric 2
                      to 10.0.0.5 via fe-1/2/2.0
                    > to 10.1.0.2 via fe-1/2/1.0
10.0.0.4/30        *[Direct/0] 23:53:21
                    > via fe-1/2/2.0
10.0.0.6/32        *[Local/0] 23:53:23
                      Local via fe-1/2/2.0
10.0.0.20/30       *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
10.1.0.0/30        *[Direct/0] 23:53:19
                    > via fe-1/2/1.0
10.1.0.1/32        *[Local/0] 23:53:23
                      Local via fe-1/2/1.0
10.3.0.0/30        *[Direct/0] 23:53:22
                    > via fe-1/2/0.0
10.3.0.2/32        *[Local/0] 23:53:23
                      Local via fe-1/2/0.0
10.3.0.4/30        *[Direct/0] 23:53:23
                    > via fe-1/2/3.0
                    [BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
                    [BGP/170] 23:52:13, localpref 100, from 192.168.0.1
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.1.0.2 via fe-1/2/1.0
10.3.0.6/32        *[Local/0] 23:53:23
                      Local via fe-1/2/3.0
10.3.0.48/30       *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
172.16.8.0/21      *[BGP/170] 00:12:26, localpref 100, from 192.168.0.1
                      AS path: 64514 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
                    [BGP/170] 00:12:26, localpref 100
                      AS path: 64515 64514 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.16.0/21     *[BGP/170] 02:03:28, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.24.0/25     *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
```

```
172.16.24.128/25   *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.25.0/26     *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.25.64/26    *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.32.0/21     *[Aggregate/130] 22:40:38
                      Reject
172.16.32.0/24     *[BGP/170] 22:45:44, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
172.16.33.0/24     *[BGP/170] 22:45:44, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
172.16.34.0/24     *[Static/5] 22:40:38
                      Reject
172.16.35.0/24     *[Static/5] 22:40:38
                      Reject
172.16.36.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.37.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.38.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.39.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.40.0/25     *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.40.128/25   *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.41.0/25     *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.41.128/25   *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.44.0/23     *[Aggregate/130] 22:40:38
                      Reject
172.16.44.0/26     *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
```

```
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.64/26    *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.128/26   *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.192/26   *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
192.168.0.1/32     *[OSPF/10] 23:52:25, metric 1
                    > to 10.1.0.2 via fe-1/2/1.0
192.168.0.2/32     *[Direct/0] 23:53:23
                    > via lo0.0
192.168.0.3/32     *[OSPF/10] 23:52:30, metric 1
                    > to 10.0.0.5 via fe-1/2/2.0
192.168.0.5/32     *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
224.0.0.5/32       *[OSPF/10] 23:53:25, metric 1
                      MultiRecv
```

Verifying the Routes on Device ISP-3

Purpose On Device ISP-3, check the routes in the routing table.

Action `user@ISP-3> show route`

```
inet.0: 40 destinations, 41 routes (40 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Aggregate/130] 23:53:57, metric2 1
                    > to 10.0.0.2 via fe-1/2/0.0
                    [BGP/170] 22:46:17, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
10.0.0.0/30        *[Direct/0] 23:53:52
                    > via fe-1/2/0.0
10.0.0.1/32        *[Local/0] 23:53:53
                      Local via fe-1/2/0.0
10.0.0.4/30        *[Direct/0] 23:53:54
                    > via fe-1/2/2.0
10.0.0.5/32        *[Local/0] 23:53:54
                      Local via fe-1/2/2.0
10.0.0.8/30        *[Direct/0] 23:53:53
                    > via fe-1/2/1.0
10.0.0.9/32        *[Local/0] 23:53:53
                      Local via fe-1/2/1.0
10.0.0.20/30       *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
10.1.0.0/30        *[OSPF/10] 23:53:03, metric 2
                    > to 10.0.0.6 via fe-1/2/2.0
                      to 10.0.0.2 via fe-1/2/0.0
10.1.0.4/30        *[Direct/0] 23:53:54
                    > via fe-1/2/3.0
10.1.0.5/32        *[Local/0] 23:53:54
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:52:46, localpref 100, from 192.168.0.1
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.0.0.2 via fe-1/2/0.0
10.3.0.48/30       *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.8.0/21      *[BGP/170] 00:12:59, localpref 100, from 192.168.0.1
                      AS path: 64514 I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
172.16.16.0/21     *[BGP/170] 02:04:01, localpref 100, from 192.168.0.2
                      AS path: 64515 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.24.0/25     *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.24.128/25   *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.25.0/26     *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.25.64/26    *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.32.0/24     *[BGP/170] 22:46:17, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
```

```
172.16.33.0/24     *[BGP/170] 22:46:17, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
172.16.34.0/24     *[BGP/170] 22:41:11, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.35.0/24     *[BGP/170] 22:41:11, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.36.0/24     *[Static/5] 22:41:11
                      Reject
172.16.37.0/24     *[Static/5] 22:41:11
                      Reject
172.16.38.0/24     *[Static/5] 22:41:11
                      Reject
172.16.39.0/24     *[Static/5] 22:41:11
                      Reject
172.16.40.0/25     *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.40.128/25   *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.41.0/25     *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.41.128/25   *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.44.0/26     *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.64/26    *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.128/26   *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.192/26   *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
192.168.0.1/32     *[OSPF/10] 23:53:03, metric 1
                    > to 10.0.0.2 via fe-1/2/0.0
192.168.0.2/32     *[OSPF/10] 23:53:03, metric 1
                    > to 10.0.0.6 via fe-1/2/2.0
192.168.0.3/32     *[Direct/0] 23:53:54
                    > via lo0.0
192.168.0.5/32     *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
224.0.0.5/32       *[OSPF/10] 23:53:58, metric 1
                      MultiRecv
```

Verifying the Routes on Device Exchange-1

Purpose On Device Exchange-1, check the routes in the routing table.

Action `user@Exchange-1> show route`

```
inet.0: 23 destinations, 24 routes (23 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.20/30       *[BGP/170] 23:53:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
10.2.0.4/30        *[Direct/0] 23:54:23
                    > via fe-1/2/3.0
10.2.0.5/32        *[Local/0] 23:54:29
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:53:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
10.3.0.40/30       *[Direct/0] 23:54:27
                    > via fe-1/2/2.0
10.3.0.42/32       *[Local/0] 23:54:29
                      Local via fe-1/2/2.0
10.3.0.44/30       *[Direct/0] 23:54:29
                    > via fe-1/2/1.0
10.3.0.45/32       *[Local/0] 23:54:29
                      Local via fe-1/2/1.0
172.16.8.0/21      *[Static/5] 00:13:31
                      Reject
172.16.16.0/21     *[BGP/170] 02:04:33, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.24.0/25     *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.24.128/25   *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.25.0/26     *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.25.64/26    *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.32.0/21     *[BGP/170] 22:46:49, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.6 via fe-1/2/3.0
                     [BGP/170] 22:41:43, localpref 100
                      AS path: 64515 64510 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.40.0/22     *[BGP/170] 22:46:49, localpref 100
                      AS path: 64510 64511 I, validation-state: unverified
                    > to 10.2.0.6 via fe-1/2/3.0
172.16.44.0/23     *[BGP/170] 22:41:43, localpref 100
                      AS path: 64515 64510 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.0/26     *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.64/26    *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified
```

```
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.128/26    *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.192/26    *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
192.168.0.5/32      *[BGP/170] 23:53:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
192.168.0.6/32      *[Direct/0] 23:54:29
                    > via lo0.0
```

Verifying the Routes on Device Exchange-2

Purpose On Device Exchange-2, check the routes in the routing table.

Action

```
user@Exchange-2> show route
inet.0: 24 destinations, 26 routes (23 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.20/30       *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.0/30        *[Direct/0] 23:54:57
                    > via fe-1/2/0.0
10.3.0.1/32        *[Local/0] 23:54:57
                      Local via fe-1/2/0.0
10.3.0.4/30        *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.40/30       *[Direct/0] 23:54:57
                    > via fe-1/2/2.0
10.3.0.41/32       *[Local/0] 23:54:57
                      Local via fe-1/2/2.0
10.3.0.48/30       *[Direct/0] 23:54:57
                    > via fe-1/2/1.0
                    [BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.49/32       *[Local/0] 23:54:57
                      Local via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:14:01, localpref 100
                      AS path: 64514 I, validation-state: unverified
                    > to 10.3.0.42 via fe-1/2/2.0
172.16.16.0/21     *[Static/5] 02:05:03
                      Reject
172.16.24.0/25     *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.24.128/25   *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.25.0/26     *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.25.64/26    *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.32.0/21     *[BGP/170] 22:42:13, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.2 via fe-1/2/0.0
                    [BGP/170] 22:47:19, localpref 100
                      AS path: 64514 64510 I, validation-state: unverified
                    > to 10.3.0.42 via fe-1/2/2.0
172.16.40.0/22     *[BGP/170] 22:47:19, localpref 100
                      AS path: 64514 64510 64511 I, validation-state: unverified

                    > to 10.3.0.42 via fe-1/2/2.0
172.16.44.0/23     *[BGP/170] 22:42:13, localpref 100
                      AS path: 64510 64512 I, validation-state: unverified
                    > to 10.3.0.2 via fe-1/2/0.0
172.16.44.0/26     *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.44.64/26    *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
```

```
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.44.128/26    *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.44.192/26    *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
192.168.0.5/32      *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
192.168.0.7/32      *[Direct/0] 23:54:57
                    > via lo0.0
```

Meaning On Device Exchange-2, the default route 0/0 is hidden because the next hop for the route is its own interface to Device Private-Peer-2, from which the route was received. The route is hidden to avoid a loop.

Verifying the Routes on Device Private-Peer-1

Purpose On Device Private-Peer-1, check the routes in the routing table.

Action

```
user@Private-Peer-1> show route

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

10.2.0.0/30        *[Direct/0] 23:58:57
                    > via fe-1/2/2.0
10.2.0.1/32        *[Local/0] 5d 21:34:22
                      Local via fe-1/2/2.0
10.3.0.44/30       *[Direct/0] 23:59:02
                    > via fe-1/2/1.0
10.3.0.46/32       *[Local/0] 1d 03:19:52
                      Local via fe-1/2/1.0
172.16.32.0/24     *[BGP/170] 22:51:22, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.33.0/24     *[BGP/170] 22:51:22, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.34.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.35.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.36.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.37.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.38.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.39.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
192.168.0.4/32     *[Direct/0] 5d 21:34:22
                    > via lo0.0
```

Verifying the Routes on Device Private-Peer-2

Purpose On Device Private-Peer-2, check the routes in the routing table.

Action `user@Private-Peer-2> show route`

```
inet.0: 29 destinations, 29 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Aggregate/130] 1d 02:13:28
                    > to 10.3.0.49 via fe-1/2/1.0
10.0.0.20/30       *[Direct/0] 1d 00:00:53
                    > via fe-1/2/0.0
10.0.0.22/32       *[Local/0] 4d 23:51:14
                      Local via fe-1/2/0.0
10.3.0.4/30        *[Direct/0] 23:59:36
                    > via fe-1/2/3.0
10.3.0.5/32        *[Local/0] 5d 21:34:57
                      Local via fe-1/2/3.0
10.3.0.48/30       *[Direct/0] 23:59:35
                    > via fe-1/2/1.0
10.3.0.50/32       *[Local/0] 1d 03:20:27
                      Local via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:18:39, localpref 100
                      AS path: 64515 64514 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.16.0/21     *[BGP/170] 02:09:41, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.24.0/25     *[Static/5] 23:14:04
                      Reject
172.16.24.128/25   *[Static/5] 23:14:04
                      Reject
172.16.25.0/26     *[Static/5] 23:14:04
                      Reject
172.16.25.64/26    *[Static/5] 23:14:04
                      Reject
172.16.32.0/21     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64515 64510 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.32.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.33.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.34.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.35.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.36.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.37.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.38.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.39.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
```

```
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.40.0/22      *[BGP/170] 22:51:57, localpref 100
                      AS path: 64515 64514 64510 64511 I, validation-state:
unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.44.0/23      *[BGP/170] 22:46:51, localpref 100
                      AS path: 64515 64510 64512 I, validation-state: unverified

                    > to 10.3.0.49 via fe-1/2/1.0
172.16.44.0/26      *[BGP/170] 23:05:32, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.21 via fe-1/2/0.0
172.16.44.64/26     *[BGP/170] 23:05:32, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.21 via fe-1/2/0.0
172.16.44.128/26    *[BGP/170] 23:05:32, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.21 via fe-1/2/0.0
172.16.44.192/26    *[BGP/170] 23:05:32, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.21 via fe-1/2/0.0
192.168.0.5/32      *[Direct/0] 5d 21:34:57
                    > via lo0.0
```

Related Documentation

- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring Routing Policy Prefix Lists on page 261

Example: 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 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 on page 198
- Overview on page 198
- Configuration on page 198
- Verification on page 202

Requirements

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

Overview

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.

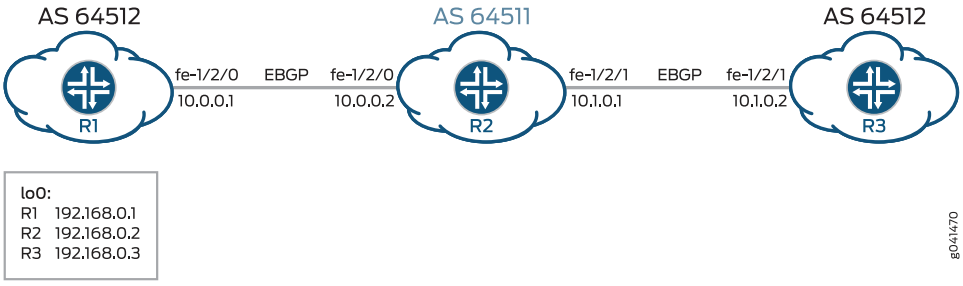
Device R2 receives these loopback interface routes, and the **advertise peer-as** statement allows Device R2 to advertise them. Specifically, Device R1 sends the 192.168.0.1 route to Device R2, and because Device R2 has the **advertise peer-as** configured, Device R2 can send the 192.168.0.1 route to 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 20 on page 198 shows the sample network.

Figure 20: BGP Topology for advertise-peer-as



"CLI Quick Configuration" on page 198 shows the configuration for all of the devices in Figure 20 on page 198.

The section "Step-by-Step Procedure" on page 199 describes the steps on Device R1 and Device R2.

Configuration

CLI Quick Configuration

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

Device 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 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 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 300
```

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 advertise-peer-as
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 300
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 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 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 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
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 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 200
   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 300
```

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 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 BGP.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 300
user@R2# set neighbor 10.1.0.2 peer-as 300
```

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 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.

Device R1

```
user@R1# show interfaces
fe-1/2/0 {
  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 200;
    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 300;
```


Device R2

```
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;
    advertise-peer-as;
    export send-direct;
    neighbor 10.0.0.1 {
      peer-as 300;
    }
    neighbor 10.1.0.2 {
      peer-as 300;
    }
  }
}

user@R2# show policy-options
policy-statement send-direct {
  term 1 {
    from protocol direct;
    then accept;
  }
}

user@R2# show routing-options
autonomous-system 200;
```

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

Verification

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                                300 I
   * 192.168.0.1/32          10.0.0.1                                300 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                                    300 I
   * 192.168.0.2/32          Self                                    I
   * 192.168.0.3/32          10.1.0.2                                300 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                              200 I
  10.1.0.0/30            10.1.0.1                              200 I
* 192.168.0.2/32         10.1.0.1                              200 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                                200 I
  10.1.0.0/30            10.1.0.1                                200 I
* 192.168.0.1/32         10.1.0.1                                200 300 I
* 192.168.0.2/32         10.1.0.1                                200 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.

Related Documentation

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

Example: Configuring BGP to Advertise the Best External Route to Internal Peers

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 65535:65284;

- Requirements on page 206
- Overview on page 206
- Configuration on page 207
- Verification on page 210

Requirements

Junos OS 9.3 or later is required.

Overview

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.

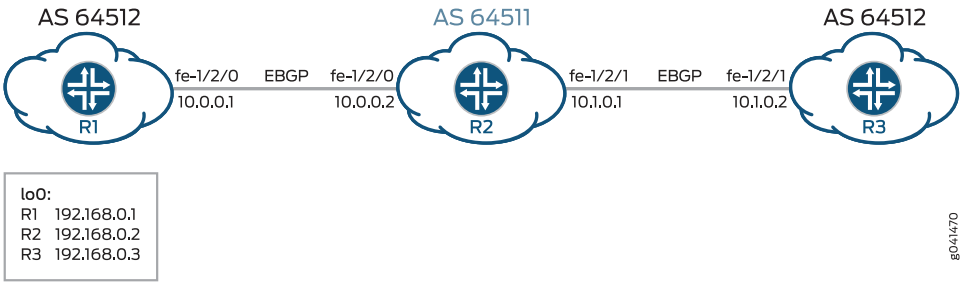
When the **advertise-external** statement is configured on Device R2 on the session toward Device R3, 172.16.6.0/24 is advertised by Device R2 toward Device R3.

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.

Topology

Figure 21 on page 206 shows the sample network.

Figure 21: BGP Topology for advertise-external



"CLI Quick Configuration" on page 207 shows the configuration for all of the devices in Figure 21 on page 206.

The section "Step-by-Step Procedure" on page 208 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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
```

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 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

Confirm that the configuration is working properly.

- Verifying the BGP Active Path on page 210
- Verifying the External Route Advertisement on page 210
- Verifying the Route on Device R3 on page 210
- Experimenting with the conditional Option on page 211

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.

Related Documentation

- Example: Setting BGP to Advertise Inactive Routes on page 212
- *Understanding BGP Path Selection*

Example: Setting BGP to Advertise Inactive Routes

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.

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 65535:65284;
```


- Requirements on page 213
- Overview on page 213
- Configuration on page 214
- Verification on page 216

Requirements

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

Overview

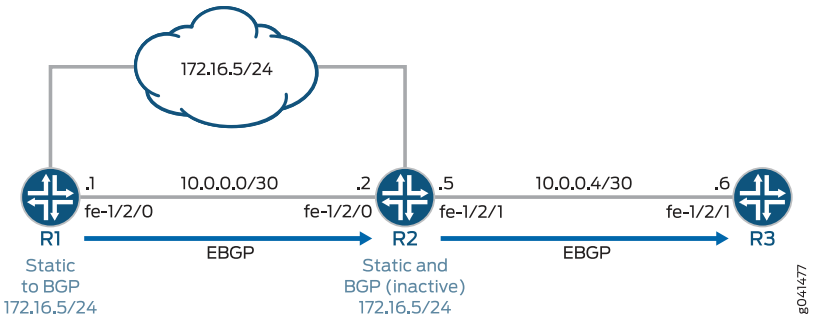
In this example, Device R2 has two external BGP (EBGP) peers, Device R1 and Device R3.

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 22 on page 213 shows the sample network.

Figure 22: BGP Topology for advertise-inactive



"CLI Quick Configuration" on page 214 shows the configuration for all of the devices in Figure 22 on page 213.

The section "Step-by-Step Procedure" on page 214 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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
```

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 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

Confirm that the configuration is working properly.

- Verifying the BGP Active Path on page 216
- Verifying the External Route Advertisement on page 217
- Verifying the Route on Device R3 on page 217
- Experimenting with the advertise-inactive Statement on page 217

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.

Related Documentation

- Example: Configuring BGP to Advertise the Best External Route to Internal Peers on page 204
- *Understanding BGP Path Selection*

Example: Rejecting Known Invalid Routes

This example shows how to create a routing policy that prevents known invalid routes from being accepted into a routing table. By default, Junos OS rejects known invalid routes (such as martian routes). This example shows how to implement an extra level of protection that takes effect when the default martian route policy is accidentally or intentionally overridden.

- Requirements on page 218
- Overview on page 218
- Configuration on page 219
- Verification on page 222

Requirements

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

Overview

In this example, you create a policy called rejectpolicy1 that rejects routes with a mask of /8 and greater (/8, /9, /10, and so on) that have the first 8 bits set to 0. This policy also accepts routes less than 8 bits in length by creating a mask of 0/0 up to /7.

The example shows two devices in two different autonomous systems (ASs) with an external BGP (EBGP) peering session. Device R1 has multiple static routes configured and a routing policy that sends static routes through BGP. The static routes are as follows:

```
user@R1# show routing-options
static {
   route 0.0.0.0/8 reject;
   route 0.0.0.0/7 reject;
   route 0.0.0.0/0 reject;
}
```

Junos OS ordinarily ignores 0/8 and longer IP addresses. Therefore, to make this example work it is necessary to override the default behavior of Junos OS. To do this, both Device R1 and Device R2 require the following setting:

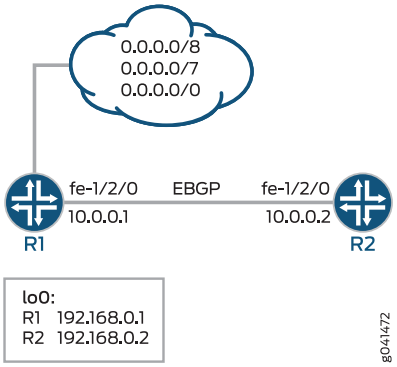
```
user@R1(and on R2)# show routing-options
rib inet.0 {
   martians {
      0.0.0.0/8 orlonger allow;
   }
}
```

This setting is required on both devices. Without this setting on Device R1, Device R1 does not send 0/8 and longer IP addresses to Device R2. With the setting on Device R1 but omitted on Device R2, Device R2 ignores incoming 0/8 and longer IP addresses.

Topology

Figure 23 on page 219 shows the sample network.

Figure 23: BGP Invalid Routes Topology



"CLI Quick Configuration" on page 219 shows the configuration for all of the devices in Figure 23 on page 219.

The section "Step-by-Step Procedure" on page 220 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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 rib inet.0 martians 0.0.0.0/8 orlonger allow
set routing-options static route 0.0.0.0/8 reject
set routing-options static route 0.0.0.0/7 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 lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext import rejectpolicy1
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set policy-options policy-statement rejectpolicy1 term rejectterm1 from route-filter
  0.0.0.0/0 upto /7 accept
set policy-options policy-statement rejectpolicy1 term rejectterm1 from route-filter
  0.0.0.0/8 orlonger reject
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 rib inet.0 martians 0.0.0.0/8 orlonger allow
set routing-options autonomous-system 200
```

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 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 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. Add the disallowed 0/8 and longer martian routes to be installed in the IPv4 unicast routing table.

   ```
   [edit routing-options rib inet.0]
   user@R2# set martians 0.0.0.0/8 orlonger allow
   ```

5. Configure a routing policy that specifies which routes to accept and which routes to reject.

   ```
   [edit policy-options policy-statement rejectpolicy1 term rejectterm1]
   user@R2# set from route-filter 0.0.0.0/0 upto /7 accept
   user@R2# set from route-filter 0.0.0.0/8 orlonger reject
   ```

6. Configure the external peering with Device R2.

   ```
   [edit protocols bgp group ext]
   user@R2# set type external
   user@R2# set neighbor 10.0.0.1 peer-as 100
   ```

7. Apply the routing policies.

```
[edit protocols bgp group ext]
user@R2# set import rejectpolicy1
user@R2# set export send-direct
```

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;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    group ext {
        type external;
        import rejectpolicy1;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement rejectpolicy1 {
    term rejectterm1 {
        from {
            route-filter 0.0.0.0/0 upto /7 accept;
            route-filter 0.0.0.0/8 orlonger reject;
        }
    }
}
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R2# show routing-options
rib inet.0 {
    martians {
```

```
        0.0.0.0/8 orlonger allow;
      }
    }
    autonomous-system 200;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes Sent from Device R2 on page 222
- Checking Device R2's Routing Table on page 222

Verifying the Routes Sent from Device R2

Purpose Make sure that Device R1 is sending the 0/8 prefix to Device R2.

Action From operational mode, enter the **show route advertising-protocol bgp** command.

```
user@R1> show route advertising-protocol bgp 10.0.0.2

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               Self                                    I
* 0.0.0.0/7               Self                                    I
* 0.0.0.0/8               Self                                    I
* 10.0.0.0/30             Self                                    I
* 192.168.0.1/32          Self                                    I
```

Meaning The output shows that Device R1 is sending all of the static routes, including the 0/8 routes, to Device R2.

Checking Device R2's Routing Table

Purpose Make sure that the 0/0 and 0/7 prefixes are accepted, and the 0/8 route is rejected.

Action From operational mode, enter the **show route protocols bgp** command.

```
user@R2> show route protocol bgp
inet.0: 7 destinations, 8 routes (6 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 02:42:25, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
0.0.0.0/7          *[BGP/170] 02:42:25, 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/170] 1d 01:01:51, 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/170] 1d 01:01:51, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
```

Meaning The output shows that on Device R2, the 0/0 and 0/7 routes are accepted into the routing table, and the 0/8 route is missing, as intended.

Related Documentation

- *Understanding Martian Addresses*
- *Example: Configuring Martian Addresses*
- Route Filter Match Conditions on page 115

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

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 on page 223
- Overview on page 223
- Configuration on page 224
- Verification on page 227

Requirements

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

Overview

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

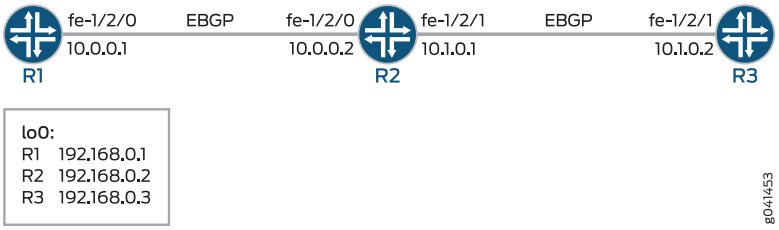
In the sample network, Device R1 and Device R3 have EBGP sessions with Device R2.

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 24 on page 224 shows the sample network.

Figure 24: BGP Preference Value Topology



"CLI Quick Configuration" on page 224 shows the configuration for all of the devices in Figure 24 on page 224.

The section "Step-by-Step Procedure" on page 225 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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
```

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 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

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.

Related Documentation

- *Route Preferences Overview*
- *Understanding External BGP Peering Sessions*
- *BGP Configuration Overview*

CHAPTER 17

Route Filters

- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF on page 240
- Example: Configuring the MED Using Route Filters on page 245
- Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters on page 257

Example: Configuring Policy Chains and Route Filters

A *policy chain* is the application of multiple policies within a specific section of the configuration. A *route filter* is a collection of match prefixes.

- Requirements on page 229
- Overview on page 229
- Configuration on page 231
- Verification on page 237

Requirements

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

Overview

An example of a policy chain applied to BGP is as follows:

```
user@R1# show protocols bgp
group int {
   type internal;
   local-address 192.168.0.1;
   export [ adv-statics adv-large-aggregates adv-small-aggregates ];
   neighbor 192.168.0.2;
   neighbor 192.168.0.3;
}
```

The **adv-statics**, **adv-large-aggregates**, and **adv-small-aggregates** policies, in addition to the default BGP policy, make up the policy chain applied to the BGP peers of Device R1. Two of the policies demonstrate route filters with different match types. The other policy matches all static routes, so no route filter is needed.

```
user@R1# show policy-options
policy-statement adv-large-aggregates {
   term between-16-and-18 {
      from {
         protocol aggregate;
         route-filter 172.16.0.0/16 upto /18;
      }
      then accept;
   }
}
policy-statement adv-small-aggregates {
   term between-19-and-24 {
      from {
         protocol aggregate;
         route-filter 172.16.0.0/16 prefix-length-range /19-/24;
      }
      then accept;
   }
}
policy-statement adv-statics {
   term statics {
      from protocol static;
      then accept;
   }
}
```

Optionally, you can convert this policy chain into a single multiterm policy for the internal BGP (IBGP) peers. If you do this, one of the advantages of a policy chain is lost—the ability to reuse policies for different purposes.

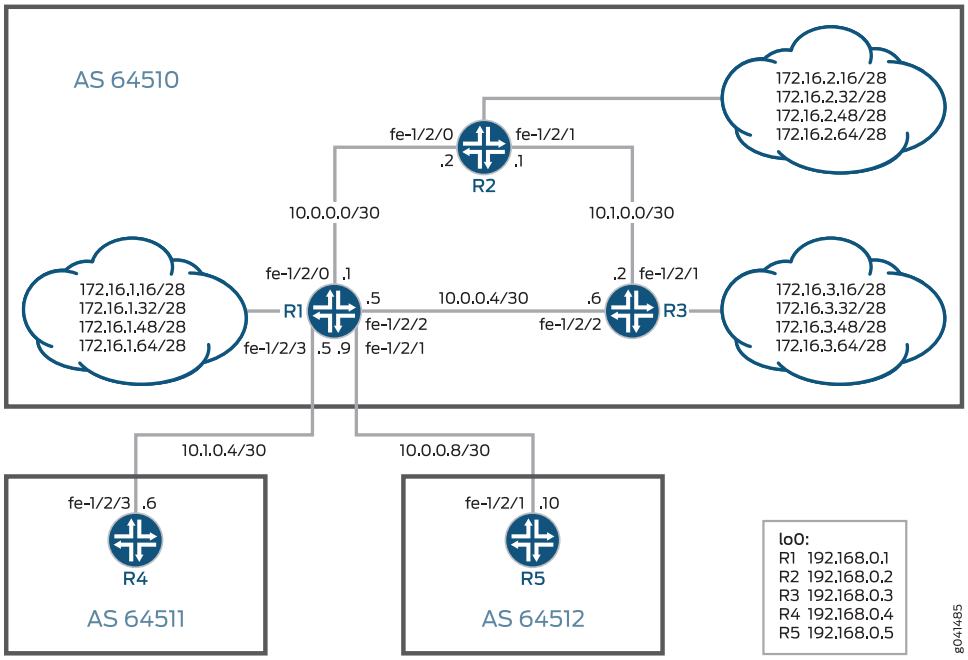
Figure 25 on page 231 displays Device R1 in AS 64510 with its IBGP peers, Device R2 and Device R3. Device R1 also has external BGP (EBGP) connections to Device R4 in AS 64511 and Device R5 in AS 64512. The current administrative policy within AS 64510 is to send the customer static routes only to other IBGP peers. Any EBGP peer providing transit service only receives aggregate routes with mask lengths smaller than 18 bits. Any EBGP peer providing peering services receives all customer routes and all aggregates whose mask length is larger than 19 bits. Each portion of these administrative policies is configured in a separate routing policy within the **[edit policy-opitons]** configuration hierarchy. These policies provide the administrators of AS 64510 with multiple configuration options for advertising routes to peers.

Device R4 is providing transit service to AS 64510, which allows the AS to advertise its assigned routing space to the Internet. On the other hand, the peering service provided by Device R5 allows AS 64510 to route traffic directly between the autonomous systems (ASs) for all customer routes.

Topology

Figure 25 on page 231 shows the sample network.

Figure 25: BGP Topology for Policy Chains



"CLI Quick Configuration" on page 231 shows the configuration for all of the devices in Figure 25 on page 231.

The section "Step-by-Step Procedure" on page 233 describes the steps on Device R1.

Configuration

CLI Quick Configuration

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

Device 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 fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/1 unit 0 description to_R5
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int export adv-statics
set protocols bgp group int export adv-large-aggregates
set protocols bgp group int export adv-small-aggregates
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 export adv-large-aggregates
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
```

```
set protocols bgp group to_64512 type external
set protocols bgp group to_64512 export adv-small-aggregates
set protocols bgp group to_64512 export adv-statics
set protocols bgp group to_64512 neighbor 10.0.0.9 peer-as 64512
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement adv-large-aggregates term between-16-and-18 from
  protocol aggregate
set policy-options policy-statement adv-large-aggregates term between-16-and-18 from
  route-filter 172.16.0.0/16 upto /18
set policy-options policy-statement adv-large-aggregates term between-16-and-18 then
  accept
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  from protocol aggregate
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  from route-filter 172.16.0.0/16 prefix-length-range /19-/24
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  then accept
set policy-options policy-statement adv-statics term statics from protocol static
set policy-options policy-statement adv-statics term statics then accept
set routing-options static route 172.16.1.16/28 discard
set routing-options static route 172.16.1.32/28 discard
set routing-options static route 172.16.1.48/28 discard
set routing-options static route 172.16.1.64/28 discard
set routing-options aggregate route 172.16.0.0/16
set routing-options aggregate route 172.16.1.0/24
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

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 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
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 export send-static-aggregate
set protocols bgp group int neighbor 192.168.0.3
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 protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static-aggregate term 1 from protocol static
set policy-options policy-statement send-static-aggregate term 1 from protocol aggregate
set policy-options policy-statement send-static-aggregate term 1 then accept
set routing-options static route 172.16.2.16/28 discard
set routing-options static route 172.16.2.32/28 discard
set routing-options static route 172.16.2.48/28 discard
set routing-options static route 172.16.2.64/28 discard
set routing-options aggregate route 172.16.2.0/24
set routing-options aggregate route 172.16.0.0/16
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
```

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
set interfaces fe-1/2/2 unit 0 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 neighbor 192.168.0.1 export send-static-aggregate
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
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 send-static-aggregate from protocol static
set policy-options policy-statement send-static-aggregate from protocol aggregate
set policy-options policy-statement send-static-aggregate then accept
set routing-options static route 172.16.3.16/28 discard
set routing-options static route 172.16.3.32/28 discard
set routing-options static route 172.16.3.48/28 discard
set routing-options static route 172.16.3.64/28 discard
set routing-options aggregate route 172.16.0.0/16
set routing-options aggregate route 172.16.3.0/24
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Device R5

```
set interfaces fe-1/2/1 unit 0 description to_R1
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols bgp group ext type external
set protocols bgp group ext neighbor 10.0.0.10 peer-as 64510
set routing-options autonomous-system 64512
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 description to_R2
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set fe-1/2/2 unit 0 description to_R3
user@R1# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@R1# set fe-1/2/3 unit 0 description to_R4
user@R1# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30
```

```
user@R1# set fe-1/2/1 unit 0 description to_R5
user@R1# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30


user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the IBGP connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Apply the export policies for the internal peers.

```
[edit protocols bgp group int]
user@R1# set export adv-statics
user@R1# set export adv-large-aggregates
user@R1# set export adv-small-aggregates
```

4. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set neighbor 10.1.0.6 peer-as 64511
```

5. Apply the export policy for Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set export adv-large-aggregates
```

6. Configure the EBGP connection to Device R5.

```
[edit protocols bgp group to_64512]
user@R1# set type external
user@R1# set neighbor 10.0.0.9 peer-as 64512
```

7. Apply the export policies for Device R5.

```
[edit protocols bgp group to_64512]
user@R1# set export adv-small-aggregates
user@R1# set export adv-statics
```

8. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

9. Configure the routing policies.

```
[edit policy-options policy-statement adv-large-aggregates term between-16-and-18]
user@R1# set from protocol aggregate
user@R1# set from route-filter 172.16.0.0/16 upto /18
user@R1# set then accept


[edit policy-options policy-statement adv-small-aggregates term
  between-19-and-24]
user@R1# set from protocol aggregate
user@R1# set from route-filter 172.16.0.0/16 prefix-length-range /19-/24
```

```
user@R1# set then accept

[edit policy-options policy-statement adv-statics term statics]
user@R1# set from protocol static
user@R1# set then accept
```

10. Configure the static and aggregate routes.

```
[edit routing-options static]
user@R1# set route 172.16.1.16/28 discard
user@R1# set route 172.16.1.32/28 discard
user@R1# set route 172.16.1.48/28 discard
user@R1# set route 172.16.1.64/28 discard

[edit routing-options aggregate]
user@R1# set route 172.16.0.0/16
user@R1# set route 172.16.1.0/24
```

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

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

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_R2;
      family inet {
         address 10.0.0.1/30;
      }
   }
}
fe-1/2/2 {
   unit 0 {
      description to_R3;
      family inet {
         address 10.0.0.5/30;
      }
   }
}
fe-1/2/3 {
   unit 0 {
      description to_R4;
      family inet {
         address 10.1.0.5/30;
      }
   }
}
fe-1/2/1 {
   unit 0 {
```

```
      description to_R5;
      family inet {
        address 10.0.0.10/30;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.0.1/32;
      }
    }
  }
```

```
user@R1# show protocols
bgp {
  group int {
    type internal;
    local-address 192.168.0.1;
    export [ adv-statics adv-large-aggregates adv-small-aggregates ];
    neighbor 192.168.0.2;
    neighbor 192.168.0.3;
  }
  group to_64511 {
    type external;
    export adv-large-aggregates;
    neighbor 10.1.0.6 {
      peer-as 64511;
    }
  }
  group to_64512 {
    type external;
    export [ adv-small-aggregates adv-statics ];
    neighbor 10.0.0.9 {
      peer-as 64512;
    }
  }
}
ospf {
  area 0.0.0.0 {
    interface fe-1/2/0.0;
    interface fe-1/2/2.0;
    interface lo0.0 {
      passive;
    }
  }
}
```

```
user@R1# show policy-options
policy-statement adv-large-aggregates {
  term between-16-and-18 {
    from {
      protocol aggregate;
      route-filter 172.16.0.0/16 upto /18;
    }
    then accept;
  }
```

```
}
policy-statement adv-small-aggregates {
   term between-19-and-24 {
      from {
         protocol aggregate;
         route-filter 172.16.0.0/16 prefix-length-range /19-/24;
      }
      then accept;
   }
}
policy-statement adv-statics {
   term statics {
      from protocol static;
      then accept;
   }
}
```

```
user@R1# show routing-options
static {
   route 172.16.1.16/28 discard;
   route 172.16.1.32/28 discard;
   route 172.16.1.48/28 discard;
   route 172.16.1.64/28 discard;
}
aggregate {
   route 172.16.0.0/16;
   route 172.16.1.0/24;
}
router-id 192.168.0.1;
autonomous-system 64510;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Route Advertisement to Device R4 on page 237
- Checking Where the Longer Routes Are Originating on page 238
- Blocking the More Specific Routes on page 238
- Verifying the Route Advertisement to Device R5 on page 239

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the customer routes are advertised to Device R4.

Action

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.0.0/16           Self                                    I
* 172.16.2.0/24           Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.0/24           Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Meaning The **adv-large-aggregates** policy is applied to the peering session with Device R4 to advertise the aggregate routes with a subnet mask length between 16 and 18 bits. The 172.16.0.0/16 aggregate route is being sent as defined by the administrative policy, but a number of other routes with larger subnet masks are also being sent to Device R4.

Checking Where the Longer Routes Are Originating

Purpose On Device R1, find where the other routes are coming from.

Action

```
user@R1> show route 172.16.3.16/28

inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.3.16/28     *[BGP/170] 20:16:00, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
```

Meaning Device R1 has learned this route through its BGP session with Device R3. Because it is an active BGP route, it is automatically advertised by the BGP default policy. Remember that the default policy is always applied to the end of every policy chain. What is needed is a policy to block the more specific routes from being advertised.

Blocking the More Specific Routes

Purpose Create a policy called **not-larger-than-18** that rejects all routes within the 172.16.0.0 /16 address space that have a subnet mask length greater than or equal to 19 bits. This ensures that all aggregates with a mask between 16 and 18 bits are advertised, thus accomplishing the goal of the administrative policy.

Action

1. On Device R1, configure the **not-larger-than-18** policy.

```
[edit policy-options policy-statement not-larger-than-18 term
  reject-greater-than-18-bits]
user@R1# set from route-filter 172.16.0.0/16 prefix-length-range /19-/32
user@R1# set then reject
```

2. On Device R1, apply the policy to the peering session with Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set export not-larger-than-18
user@R1# commit
```

3. On Device R1, check which routes are advertised to Device R4.

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.0.0/16           Self                                    I
```

Meaning The policy chain is working correctly. Only the 172.16.0.0 /16 route is advertised to Device R4.

Verifying the Route Advertisement to Device R5

Purpose On Device R1, make sure that the customer routes are advertised to Device R5.

Device R5 is Device R1's EBGP peer in AS 64512. The administrative policy states that this peer receives only aggregate routes larger than 18 bits in length and all customer routes. In anticipation of encountering a problem similar to the problem on Device R4, you can create a policy called **not-smaller- than-18** that rejects all aggregates with mask lengths between 16 and 18 bits.

Action 1. On Device R2, configure an aggregate route for 172.16.128.0/17.

```
[edit routing-options aggregate]
user@R2# set route 172.16.128.0/17 discard
user@R2# commit
```

2. On Device R1, check which routes are advertised to Device R5.

```
user@R1> show route advertising-protocol bgp 10.0.0.9


inet.0: 30 destinations, 32 routes (30 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.0/24           Self                                    I
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.0/24           Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.0/24           Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
* 172.16.128.0/17         Self                                    I
```

The aggregate route 172.16.128.0/17 is advertised, in violation of the administrative policy

3. On Device R1, configure the **not-smaller-than-18** policy.

   ```
   [edit policy-options policy-statement not-smaller-than-18 term reject-less-than-18-bits]
   user@R1# set from protocol aggregate
   user@R1# set from route-filter 172.16.0.0/16 upto /18
   user@R1# set then reject
   ```

4. On Device R1, apply the policy to the peering session with Device R5.

   ```
   [edit protocols bgp group to_64512]
   user@R1# set export not-smaller-than-18
   user@R1# commit
   ```

5. On Device R1, check which routes are advertised to Device R5.

   ```
   user@R1> show route advertising-protocol bgp 10.0.0.9

   inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
     Prefix                  Nexthop              MED     Lclpref    AS path
   * 172.16.1.0/24           Self                                    I
   * 172.16.1.16/28          Self                                    I
   * 172.16.1.32/28          Self                                    I
   * 172.16.1.48/28          Self                                    I
   * 172.16.1.64/28          Self                                    I
   * 172.16.2.0/24           Self                                    I
   * 172.16.2.16/28          Self                                    I
   * 172.16.2.32/28          Self                                    I
   * 172.16.2.48/28          Self                                    I
   * 172.16.2.64/28          Self                                    I
   * 172.16.3.0/24           Self                                    I
   * 172.16.3.16/28          Self                                    I
   * 172.16.3.32/28          Self                                    I
   * 172.16.3.48/28          Self                                    I
   * 172.16.3.64/28          Self                                    I
   ```

Meaning The policy chain is working correctly. Only aggregate routes larger than 18 bits in length and all customer routes are advertised to Device R5.

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25
- Route Filter Match Conditions on page 115
- Example: Configuring Routing Policy Prefix Lists on page 261
- Example: Configuring a Policy Subroutine on page 273

Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF

This example shows how to create an OSPF import policy that prioritizes specific prefixes learned through OSPF.

- Requirements on page 241
- Overview on page 241

- Configuration on page 242
- Verification on page 244

Requirements

Before you begin:

- Configure the device interfaces.
- Configure the router identifiers for the devices in your OSPF network. See *Example: Configuring an OSPF Router Identifier*.
- Control OSPF designated router election. See *Example: Controlling OSPF Designated Router Election*.
- Configure a single-area OSPF network. See *Example: Configuring a Single-Area OSPF Network*.
- Configure a multiarea OSPF network. See *Example: Configuring a Multiarea OSPF Network*.

Overview

In a network with a large number of OSPF routes, it can be useful to control the order in which routes are updated in response to a network topology change. In Junos OS Release 9.3 and later, you can specify a priority of high, medium, or low for prefixes included in an OSPF import policy. In the event of an OSPF topology change, high priority prefixes are updated in the routing table first, followed by medium and then low priority prefixes.

OSPF import policy can only be used to set priority or to filter OSPF external routes. If an OSPF import policy is applied that results in a **reject** terminating action for a nonexternal route, then the **reject** action is ignored and the route is accepted anyway. By default, such a route is now installed in the routing table with a priority of low. This behavior prevents traffic black holes, that is, silently discarded traffic, by ensuring consistent routing within the OSPF domain.

In general, OSPF routes that are not explicitly assigned a priority are treated as priority medium, except for the following:

- Summary discard routes have a default priority of low.
- Local routes that are not added to the routing table are assigned a priority of low.
- External routes that are rejected by import policy and thus not added to the routing table are assigned a priority of low.

Any available match criteria applicable to OSPF routes can be used to determine the priority. Two of the most commonly used match criteria for OSPF are the **route-filter** and **tag** statements.

In this example, the routing device is in area 0.0.0.0, with interfaces fe-0/1/0 and fe-1/1/0 connecting to neighboring devices. You configure an import routing policy named ospf-import to specify a priority for prefixes learned through OSPF. Routes associated

with these prefixes are installed in the routing table in the order of the prefixes' specified priority. Routes matching **200.3.0.0/16 orlonger** are installed first because they have a priority of **high**. Routes matching **200.2.0.0/16 orlonger** are installed next because they have a priority of **medium**. Routes matching **200.1.0.0/16 orlonger** are installed last because they have a priority of **low**. You then apply the import policy to OSPF.

NOTE: The priority value takes effect when a new route is installed, or when there is a change to an existing route.

Configuration

CLI Quick Configuration

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

```
set interfaces fe-0/1/0 unit 0 family inet address 192.168.8.4/30
set interfaces fe-0/2/0 unit 0 family inet address 192.168.8.5/30
set policy-options policy-statement ospf-import term t1 from route-filter 200.1.0.0/16
  orlonger
set policy-options policy-statement ospf-import term t1 then priority low
set policy-options policy-statement ospf-import term t1 then accept
set policy-options policy-statement ospf-import term t2 from route-filter 200.2.0.0/16
  orlonger
set policy-options policy-statement ospf-import term t2 then priority medium
set policy-options policy-statement ospf-import term t2 then accept
set policy-options policy-statement ospf-import term t3 from route-filter 200.3.0.0/16
  orlonger
set policy-options policy-statement ospf-import term t3 then priority high
set policy-options policy-statement ospf-import term t3 then accept
set protocols ospf import ospf-import
set protocols ospf area 0.0.0.0 interface fe-0/1/0.0
set protocols ospf area 0.0.0.0 interface fe-0/2/0.0
```

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 OSPF import policy that prioritizes specific prefixes:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@host# set fe-0/1/0 unit 0 family inet address 192.168.8.4/30

   user@host# set fe-0/2/0 unit 0 family inet address 192.168.8.5/30
   ```

2. Enable OSPF on the interfaces.

NOTE: For OSPFv3, include the **ospf3** statement at the **[edit protocols]** hierarchy level.

```
[edit protocols ospf area 0.0.0.0]
user@host# set interface fe-0/1/0.0
user@host# set interface fe-0/2/0.0
```

3. Configure the policy to specify the priority for prefixes learned through OSPF.

```
[edit policy-options policy-statement ospf-import]
user@host# set term t1 from route-filter 200.1.0.0/16 orlonger
user@host# set term t1 then priority low
user@host# set term t1 then accept

user@host# set term t2 from route-filter 200.2.0.0/16 orlonger
user@host# set term t2 then priority medium
user@host# set term t2 then accept

user@host# set term t3 from route-filter 200.3.0.0/16 orlonger
user@host# set term t3 then priority high
user@host# set term t3 then accept
```

4. Apply the policy to OSPF.

```
[edit protocols ospf]
user@host# set import ospf-import
```

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

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

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols ospf**, 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 interfaces
fe-0/1/0 {
    unit 0 {
        family inet {
            address 192.168.8.4/30;
        }
    }
}
    fe-0/2/0 {
        unit 0 {
            family inet {
                address 192.168.8.5/30;
            }
        }
    }
```

```
user@host# show protocols ospf
import ospf-import;
area 0.0.0.0 {
    interface fe-0/1/0.0;
    interface fe-0/2/0.0;
}
```

```
user@host# show policy-options
policy-statement ospf-import {
  term t1 {
    from {
      route-filter 200.1.0.0/16 orlonger;
    }
    then {
      priority low;
      accept;
    }
  }
  term t2 {
    from {
      route-filter 200.2.0.0/16 orlonger;
    }
    then {
      priority medium;
      accept;
    }
  }
  term t3 {
    from {
      route-filter 200.3.0.0/16 orlonger;
    }
    then {
      priority high;
      accept;
    }
  }
}
```

To confirm your OSPFv3 configuration, enter the **show interfaces**, **show protocols ospf3**, and **show policy-options** commands.

Verification

Confirm that the configuration is working properly.

Verifying the Prefix Priority in the OSPF Routing Table

Purpose Verify the priority assigned to the prefix in the OSPF routing table.

Action From operational mode, enter the **show ospf route detail** for OSPFv2, and enter the **show ospf3 route detail** command for OSPFv3.

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25
- *OSPF Routing Policy Overview*
- Routing Policy Match Conditions on page 107 in the *Routing Policy Feature Guide for Routing Devices*
- Actions in Routing Policy Terms on page 119 in the *Routing Policy Feature Guide for Routing Devices*

Example: Configuring the MED Using Route Filters

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 on page 245
- Overview on page 245
- Configuration on page 245
- Verification on page 256

Requirements

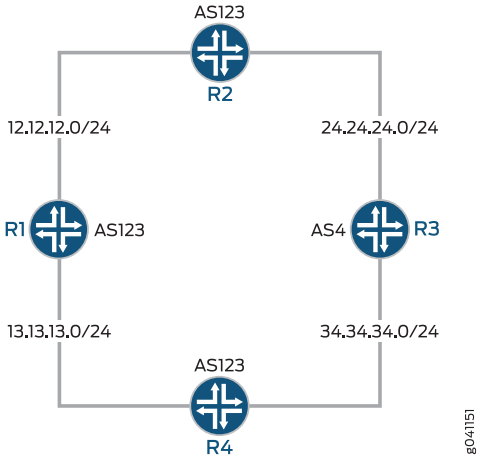
No special configuration beyond device initialization is required before you configure this example.

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 26 on page 245 shows a typical network with internal peer sessions and multiple exit points to a neighboring autonomous system (AS).

Figure 26: 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 for all routes except 144.144.144.144. For 144.144.144.144, 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

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 export med-10
set protocols bgp group external neighbor 34.34.34.3 export med-30
set protocols bgp group external neighbor 24.24.24.2 metric-out 20
set policy-options policy-statement med-10 from route-filter 144.144.144.144/32 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 *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 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 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 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 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 *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 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 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 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 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 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 *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 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 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 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 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 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 *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 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 the two MED policies.

```
[edit policy-options]
set policy-statement med-10 from route-filter 144.144.144.144/32 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 34.34.34.3 export med-10
user@R4# set neighbor 34.34.34.3 export med-30
user@R4# set neighbor 24.24.24.2 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 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 protocols
bgp {
    group external {
      type external;
      export send-direct;
      peer-as 123;
      neighbor 24.24.24.2 {
        metric-out 20;
      }
      neighbor 34.34.34.3 {
        export [ med-10 med-30 ];
      }
    }
}
```

```
user@R4# show policy-options
policy-statement med-10 {
    from {
      route-filter 144.144.144.144/32 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

Confirm that the configuration is working properly.

- Checking the Active Path from Device R1 to Device R4 on page 256
- Verifying That Device R4 Is Sending Its Routes Correctly on page 257

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] 4d 01:13:32, 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 05:36:10, 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] 4d 01:13:32, 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 05:36:10, 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] 00:06:03, 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] 00:06:03, MED 10, localpref 100, from 192.168.3.1
                      AS path: 4 I
                    > to 13.13.13.3 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 12.12.12.2 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 13.13.13.3 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 12.12.12.2 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 144.144.144.144/32. For 144.144.144.144/32, 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 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                 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.

Related Documentation

- *Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates*
- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25
- *Understanding BGP Path Selection*
- *Understanding External BGP Peering Sessions*
- *BGP Configuration Overview*

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

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

- Requirements on page 258
- Overview on page 258
- Configuration on page 259
- Verification on page 260

Requirements

This example uses Junos OS Release 10.0 or later.

Before you begin:

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

Overview

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

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

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

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

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

```
from {
  route-filter 0.0.0.0/0 exact;
}
then {
  next-hop self;
  accept;
}
```

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

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

The family qualifier is evaluated before the route-filter is evaluated. Thus, the route-filter is not evaluated if the family match fails. The same logic applies to family **inet6** and family **inet6-vpn**. The route-filter used in the **inet6** example must use an IPv6 address.

There is a potential efficiency gain in using a family qualifier because the family qualifier is tested before most other qualifiers, quickly eliminating routes from undesired families.

Configuration

CLI Quick Configuration

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

inet Example

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

Inet-vpn Example

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

inet6 Example

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

Inet6-vpn Example

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

Step-by-Step Procedure

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

To configure a flow map:

1. Configure the family qualifier.

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

2. Configure the route filter.

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

3. Configure the policy actions.

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

4. Apply the policy.

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

Results

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

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

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

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

Verification

To verify the configuration, run the following commands:

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

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 25
- Route Filter Match Conditions on page 115
- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring the MED Using Route Filters on page 245
- Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF on page 240

CHAPTER 18

Prefix Lists

- Example: Configuring Routing Policy Prefix Lists on page 261

Example: Configuring Routing Policy Prefix Lists

In Junos OS, prefix lists provide one method of defining a set of routes. Junos OS provides other methods of accomplishing the same task, such as route filters. A prefix list is a listing of IP prefixes that represent a set of routes that are used as match criteria in an applied policy. Such a list might be useful for representing a list of customer routes in your autonomous system (AS). A prefix list is given a name and is configured within the **[edit policy-options]** configuration hierarchy.

- Requirements on page 261
- Overview on page 261
- Configuration on page 264
- Verification on page 269

Requirements

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

Overview

Prefix lists are similar to a list of route filters. The functional difference between route filters and prefix lists is that you cannot specify a range using a prefix list. You can simulate a range using a prefix list by including additional prefixes in the list, or by using two prefix lists, one shorter and one longer, setting one to accept and the other to reject. You can also filter a prefix list using the **prefix-list-filter** match condition. Your choices are **exact**, **longer**, and **orlonger**.

The benefit of a prefix list over a list of route filters is seen when the prefixes are referenced in several different locations. For instance, a prefix list can be referenced in a BGP import policy, an export policy, an RPF policy, in firewall filters, in loopback filters, in setting a multicast scope, and so on.

When your list of prefixes changes, rather than trying to remember the many different locations prefixes are configured, you can instead update the prefix list, changing the prefix one time instead of multiple times. This helps to reduce the likelihood of

configuration errors, such as mistyping the address in a location or forgetting to update one or more locations.

Prefix lists also help when managing a large number of devices. You can write the various filters and policies as generically as possible, referencing prefix lists instead of specific IP addresses. The more complex logic in the filters and policies has to be written only one time, with minimal per-device and per-site customizations.

As shown in Figure 27 on page 263, each router in AS 64510 has customer routes. Device R1 assigns customer routes within the 172.16.1.0/24 subnet. Device R2 and Device R3 assign customer routes within the 172.16.2.0/24 and 172.16.3.0/24 subnets, respectively. Device R1 has been designated the central point in AS 64510 to maintain a complete list of customer routes. Device R1 has a prefix list called **customers**, as follows:

```
user@R1# show policy-options
prefix-list customers {
    172.16.1.16/28;
    172.16.1.32/28;
    172.16.1.48/28;
    172.16.1.64/28;
    172.16.2.16/28;
    172.16.2.32/28;
    172.16.2.48/28;
    172.16.2.64/28;
    172.16.3.16/28;
    172.16.3.32/28;
    172.16.3.48/28;
    172.16.3.64/28;
}
```

As you can see, the prefix list does not contain a match type for each route (as you would see with a route filter). This is an important point when using a prefix list in a policy. Routes match only if they exactly match one of the prefixes in the list. In other words, each route in the list must appear in the routing table exactly as it is configured in the prefix list.

You reference the prefix list as a match criterion within a policy like this:

```
user@R1# show policy-options
policy-statement customer-routes {
    term get-routes {
        from {
            prefix-list customers;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
```

In this example, all the routes in the **customers** prefix list appear in the routing table on Device R1. Device R2 and Device R3 export to Device R1 static routes to their customers.

As previously mentioned, you can use the **prefix-list-filter** match condition with the **exact**, **longer**, or **orlonger** match type. This provides a way to avoid the prefix list exact-match limitation of prefix lists. For example:

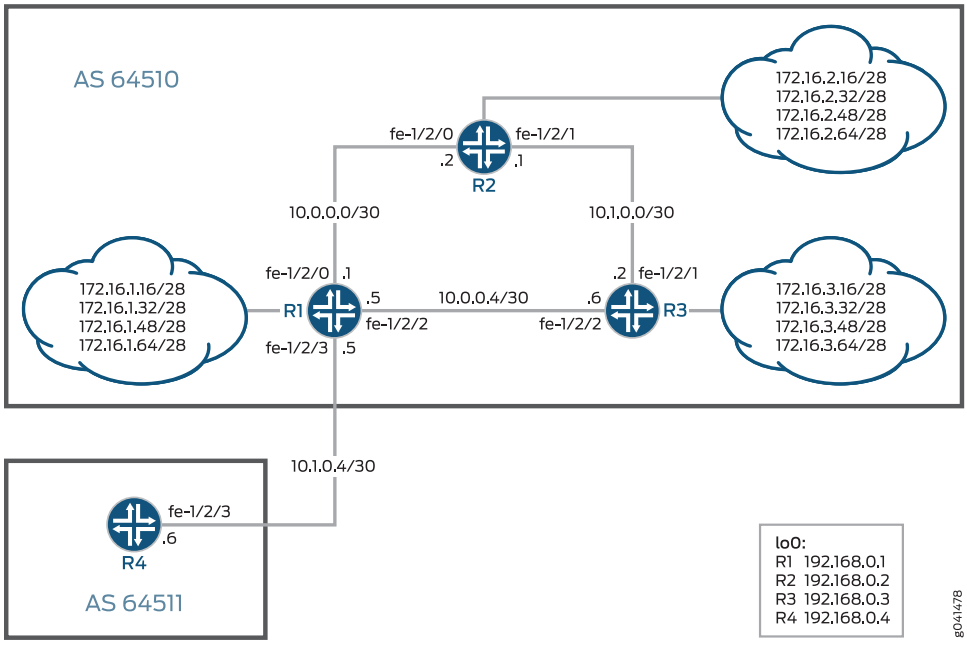
```
user@R1# show policy-options
policy-statement customer-routes {
   term get-routes {
      from {
         prefix-list-filter customers orlonger;
      }
      then accept;
   }
   term others {
      then reject;
   }
}
```

The example demonstrates the effects of both the **prefix-list** match condition and the **prefix-list-filter** match condition.

Topology

Figure 27 on page 263 shows the sample network.

Figure 27: BGP Topology for Policy Prefix Lists



"CLI Quick Configuration" on page 264 shows the configuration for all of the devices in Figure 27 on page 263.

The section "Step-by-Step Procedure" on page 265 describes the steps on Device R1.

Configuration

CLI Quick Configuration

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

Device 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 fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
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 bgp group to_64511 type external
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols bgp group to_64511 export customer-routes
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options prefix-list 64510-customers 172.6.1.16/28
set policy-options prefix-list 64510-customers 172.6.1.32/28
set policy-options prefix-list 64510-customers 172.6.1.48/28
set policy-options prefix-list 64510-customers 172.6.1.64/28
set policy-options prefix-list 64510-customers 172.6.2.16/28
set policy-options prefix-list 64510-customers 172.6.2.32/28
set policy-options prefix-list 64510-customers 172.6.2.48/28
set policy-options prefix-list 64510-customers 172.6.2.64/28
set policy-options prefix-list 64510-customers 172.6.3.16/28
set policy-options prefix-list 64510-customers 172.6.3.32/28
set policy-options prefix-list 64510-customers 172.6.3.48/28
set policy-options prefix-list 64510-customers 172.6.3.64/28
set policy-options policy-statement customer-routes term get-routes from prefix-list
  64510-customers
set policy-options policy-statement customer-routes term get-routes then accept
set policy-options policy-statement customer-routes term others then reject
set routing-options static route 172.6.1.16/28 discard
set routing-options static route 172.6.1.32/28 discard
set routing-options static route 172.6.1.48/28 discard
set routing-options static route 172.6.1.64/28 discard
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

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-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 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 int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1 export send-static
```

```
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
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 accept
set routing-options static route 172.6.2.16/28 discard
set routing-options static route 172.6.2.32/28 discard
set routing-options static route 172.6.2.48/28 discard
set routing-options static route 172.6.2.64/28 discard
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
set interfaces fe-1/2/2 unit 0 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 neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
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 send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.6.3.16/28 discard
set routing-options static route 172.6.3.32/28 discard
set routing-options static route 172.6.3.48/28 discard
set routing-options static route 172.6.3.64/28 discard
set routing-options static route 172.6.3.1/32 discard
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set interfaces fe-1/2/0 unit 0 description to_R2
user@R1# set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
```

```
user@R1# set interfaces fe-1/2/2 unit 0 description to_R3
user@R1# set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@R1# set interfaces fe-1/2/3 unit 0 description to_R4
user@R1# set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@R1# set interfaces lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the internal BGP (IBGP) connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set neighbor 10.1.0.6 peer-as 64511
user@R1# set export customer-routes
```

4. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

5. Configure the prefix list.

```
[edit policy-options prefix-list 64510-customers]
user@R1# set 172.6.1.16/28
user@R1# set 172.6.1.32/28
user@R1# set 172.6.1.48/28
user@R1# set 172.6.1.64/28
user@R1# set 172.6.2.16/28
user@R1# set 172.6.2.32/28
user@R1# set 172.6.2.48/28
user@R1# set 172.6.2.64/28
user@R1# set 172.6.3.16/28
user@R1# set 172.6.3.32/28
user@R1# set 172.6.3.48/28
user@R1# set 172.6.3.64/28
```

6. Configure the routing policy that references the prefix list as a match criterion.

```
[edit policy-options policy-statement customer-routes term get-routes]
user@R1# set from prefix-list 64510-customers
user@R1# set then accept

[edit policy-options policy-statement customer-routes term others]
user@R1# set then reject
```

7. Configure the static route to the 172.16.5.0/24 network.

```
[edit routing-options static]
user@R1# set route 172.6.1.16/28 discard
```



```
user@R1# set route 172.6.1.32/28 discard
user@R1# set route 172.6.1.48/28 discard
user@R1# set route 172.6.1.64/28 discard
```

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

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

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_R2;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        description to_R3;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        description to_R4;
        family inet {
            address 10.1.0.5/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.1;
        neighbor 192.168.0.2;
        neighbor 192.168.0.3;
    }
```

```
    group to_64511 {
      type external;
      export customer-routes;
      neighbor 10.1.0.6 {
        peer-as 64511;
      }
    }
  }
  ospf {
    area 0.0.0.0 {
      interface fe-1/2/0.0;
      interface fe-1/2/2.0;
      interface lo0.0 {
        passive;
      }
    }
  }

user@R1# show policy-options
prefix-list 64510-customers {
    172.6.1.16/28;
    172.6.1.32/28;
    172.6.1.48/28;
    172.6.1.64/28;
    172.6.2.16/28;
    172.6.2.32/28;
    172.6.2.48/28;
    172.6.2.64/28;
    172.6.3.16/28;
    172.6.3.32/28;
    172.6.3.48/28;
    172.6.3.64/28;
}
policy-statement customer-routes {
    term get-routes {
      from {
        prefix-list 64510-customers;
      }
      then accept;
    }
    term others {
      then reject;
    }
}

user@R1# show routing-options
static {
    route 172.6.1.16/28 discard;
    route 172.6.1.32/28 discard;
    route 172.6.1.48/28 discard;
    route 172.6.1.64/28 discard;
}
router-id 192.168.0.1;
autonomous-system 64510;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device R1 on page 269
- Verifying the Route Advertisement to Device R4 on page 269
- Experimenting with the prefix-list-filter Statement on page 270

Verifying the Routes on Device R1

Purpose On Device R1, check the routes in the routing table.

Action

```
user@R1> show route terse 172.16/16

inet.0: 26 destinations, 26 routes (26 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 172.16.1.16/28     S   5                        Discard
* ? 172.16.1.32/28     S   5                        Discard
* ? 172.16.1.48/28     S   5                        Discard
* ? 172.16.1.64/28     S   5                        Discard
* ? 172.16.2.1/32      B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.2.16/28     B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.2.32/28     B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.2.48/28     B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.2.64/28     B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.2.96/32     B 170        100                          I
  unverified                                      >10.0.0.2
* ? 172.16.3.1/32      B 170        100                          I
  unverified                                      >10.0.0.6
* ? 172.16.3.16/28     B 170        100                          I
  unverified                                      >10.0.0.6
* ? 172.16.3.32/28     B 170        100                          I
  unverified                                      >10.0.0.6
* ? 172.16.3.48/28     B 170        100                          I
  unverified                                      >10.0.0.6
* ? 172.16.3.64/28     B 170        100                          I
  unverified                                      >10.0.0.6
```

Meaning Device R1 has learned its own static routes (S) and the BGP routes from Devices R2 and R3 (B).

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the customer routes are advertised to Device R4.

Action

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 26 destinations, 26 routes (26 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Meaning As expected, only the routes from the customer prefix list are advertised to Device R4.

Experimenting with the prefix-list-filter Statement

Purpose See what can happen when you use **prefix-list-filter** instead of **prefix-list**.

Action

1. On Device R2, add a static route that is longer than one of the existing static routes.

   ```
   [edit routing-options static route]
   user@R2# set 172.16.2.65/32 discard
   user@R2# commit
   ```

2. On Device R1, deactivate the prefix list and configure a prefix list filter with the **orlonger** match type.

   ```
   [edit policy-options policy-statement customer-routes term get-routes]
   user@R1# deactivate from prefix-list 64510-customers
   user@R1# set from prefix-list-filter 64510-customers orlonger
   user@R1# commit
   ```

3. On Device R1, check which routes are advertised to Device R4.

   ```
   user@R1> show route advertising-protocol bgp 10.1.0.6

   inet.0: 27 destinations, 27 routes (27 active, 0 holddown, 0 hidden)
     Prefix                  Nexthop              MED     Lclpref    AS path
   * 172.16.1.16/28          Self                                    I
   * 172.16.1.32/28          Self                                    I
   * 172.16.1.48/28          Self                                    I
   * 172.16.1.64/28          Self                                    I
   * 172.16.2.16/28          Self                                    I
   * 172.16.2.32/28          Self                                    I
   * 172.16.2.48/28          Self                                    I
   * 172.16.2.64/28          Self                                    I
   * 172.16.2.65/32          Self                                    I
   * 172.16.3.16/28          Self                                    I
   * 172.16.3.32/28          Self                                    I
   * 172.16.3.48/28          Self                                    I
   * 172.16.3.64/28          Self                                    I
   ```

Meaning As expected, Device R1 is now advertising the 172.16.2.65/32 route to Device R4, even though 172.16.2.65/32 is not in the prefix list.

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 45
- Example: Configuring Policy Chains and Route Filters on page 229
- Example: Configuring a Policy Subroutine on page 273

CHAPTER 19

Subroutines

- Example: Configuring a Policy Subroutine on page 273

Example: Configuring a Policy Subroutine

This example demonstrates the use of a policy subroutine in a routing policy match condition.

- Requirements on page 273
- Overview on page 273
- Configuration on page 275
- Verification on page 280

Requirements

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

Overview

On Device R1, a policy called **main** is configured.

```
user@R1# show policy-options
policy-statement main {
    term subroutine-as-a-match {
        from policy subroutine;
        then accept;
    }
    term nothing-else {
        then reject;
    }
}
```

This main policy calls a subroutine called **subroutine**.

```
user@R1# show policy-options
policy-statement subroutine {
    term get-routes {
        from protocol static;
        then accept;
    }
```

```
    term nothing-else {
      then reject;
    }
  }
```

The router evaluates the logic of **main** in a defined manner. The match criterion of **from policy subroutine** allows the routing device to locate the subroutine. All terms of the subroutine are evaluated, in order, following the normal policy processing rules. In this example, all static routes in the routing table match the subroutine with an action of accept. This returns a true result to the original, or calling, policy which informs the device that a positive match has occurred. The actions in the calling policy are executed and the route is accepted. All other routes in the routing table do not match the subroutine and return a false result to the calling policy. The device evaluates the second term of **main** and rejects the routes.

The actions in the subroutine do not actually accept or reject a specific route. The subroutine actions are only translated into a true or a false result. Actions that modify a route's attributes, however, are applied to the route regardless of the outcome of the subroutine.

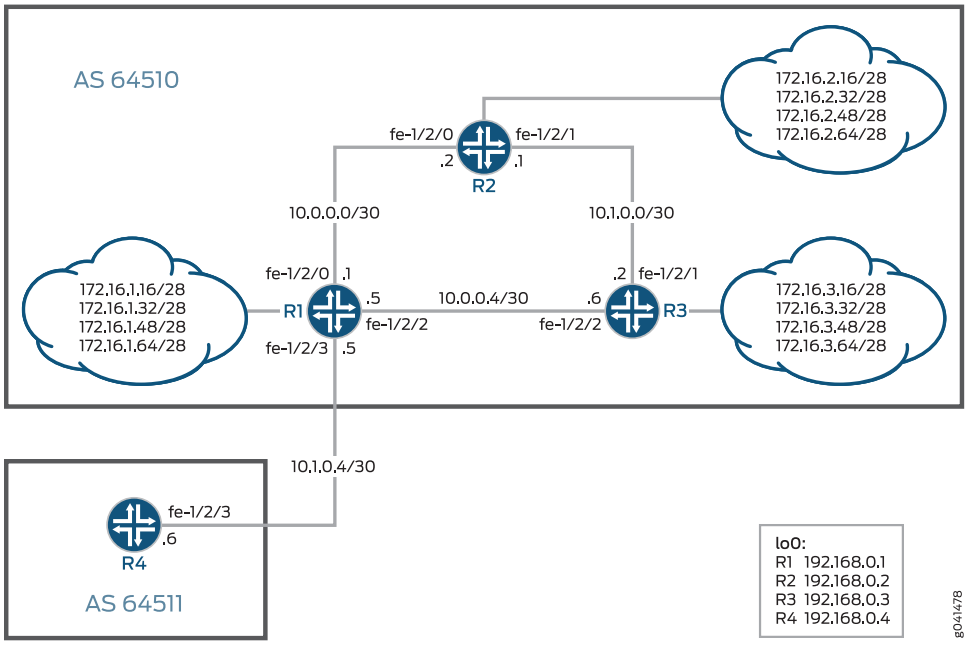
Device R1 in AS 64510 has multiple customer routes, some of which are static routes configured locally, and some of which are received from Device R2 and Device R3 through internal BGP (IBGP). AS 64510 is connected to Device R4 in AS 64511. The policy **main** is applied as an export policy in Device R1's BGP peering session with Device R4. This causes Device R1 to send only its own static routes to Device R4. Because of the policy **main**, Device R1 does not send the routes received from its internal peers, Device R2 and Device R3.

When you are working with policy subroutines, it is important to remember that the default EBGP export policy is to advertise all learned BGP routes to all EBGP peers. This default policy is in effect in the main policy and also in the subroutine. Therefore, as shown in this example, if you do not want the default EBGP export policy to take effect, you must configure a **then reject** terminating action as the final term in both the main policy and in the policy subroutine. This example demonstrates what happens when the final **then reject** term is missing either from the main policy or from the policy subroutine.

Topology

Figure 28 on page 275 shows the sample network.

Figure 28: BGP Topology for Policy Subroutine



"CLI Quick Configuration" on page 275 shows the configuration for all of the devices in Figure 28 on page 275.

The section "Step-by-Step Procedure" on page 277 describes the steps on Device R1.

Configuration

CLI Quick Configuration

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

Device 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 fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
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 bgp group to_64511 type external
set protocols bgp group to_64511 export main
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement main term subroutine-as-a-match from policy
    subroutine
set policy-options policy-statement main term subroutine-as-a-match then accept
```

```
set policy-options policy-statement main term nothing-else then reject
set policy-options policy-statement subroutine term get-routes from protocol static
set policy-options policy-statement subroutine term get-routes then accept
set policy-options policy-statement subroutine term nothing-else then reject
set routing-options static route 172.16.1.16/28 discard
set routing-options static route 172.16.1.32/28 discard
set routing-options static route 172.16.1.48/28 discard
set routing-options static route 172.16.1.64/28 discard
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

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 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
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 export send-static
set protocols bgp group int neighbor 192.168.0.3
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 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 accept
set routing-options static route 172.16.2.16/28 discard
set routing-options static route 172.16.2.32/28 discard
set routing-options static route 172.16.2.48/28 discard
set routing-options static route 172.16.2.64/28 discard
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
set interfaces fe-1/2/2 unit 0 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 neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.6
set protocols ospf area 0.0.0.0 interface fe-1/2/0.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static from protocol static
set policy-options policy-statement send-static then accept
set routing-options static route 172.16.3.16/28 discard
set routing-options static route 172.16.3.32/28 discard
set routing-options static route 172.16.3.48/28 discard
set routing-options static route 172.16.3.64/28 discard
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
```

```
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 description to_R2
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set fe-1/2/2 unit 0 description to_R3
user@R1# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@R1# set fe-1/2/3 unit 0 description to_R4
user@R1# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the internal BGP (IBGP) connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set export main
user@R1# set neighbor 10.1.0.6 peer-as 64511
```

4. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

5. Configure the policy **main**.

```
[edit policy-options policy-statement main term subroutine-as-a-match]
user@R1# set from policy subroutine
user@R1# set then accept

[edit policy-options policy-statement main term nothing-else]
user@R1# set then reject
```

6. Configure the policy **subroutine**.

```
[edit policy-options policy-statement subroutine term get-routes]
user@R1# set from protocol static
user@R1# set then accept

[edit policy-options policy-statement subroutine term nothing-else]
user@R1# set then reject
```

7. Configure the static route to the 172.16.5.0/24 network.

```
[edit routing-options static]
user@R1# set route 172.16.1.16/28 discard
user@R1# set route 172.16.1.32/28 discard
user@R1# set route 172.16.1.48/28 discard
user@R1# set route 172.16.1.64/28 discard
```

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

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

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_R2;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        description to_R3;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        description to_R4;
        family inet {
            address 10.1.0.5/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
```

```
    }
}

user@R1# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.1;
        neighbor 192.168.0.2;
        neighbor 192.168.0.3;
    }
    group to_64511 {
        type external;
        export main;
        neighbor 10.1.0.6 {
            peer-as 64511;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.0;
        interface fe-1/2/2.0;
        interface lo0.0 {
            passive;
        }
    }
}

user@R1# show policy-options
policy-statement main {
    term subroutine-as-a-match {
        from policy subroutine;
        then accept;
    }
    term nothing-else {
        then reject;
    }
}
policy-statement subroutine {
    term get-routes {
        from protocol static;
        then accept;
    }
    term nothing-else {
        then reject;
    }
}

user@R1# show routing-options
static {
    route 172.6.1.16/28 discard;
    route 172.6.1.32/28 discard;
    route 172.6.1.48/28 discard;
    route 172.6.1.64/28 discard;
}
router-id 192.168.0.1;
```

```
autonomous-system 64510;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device R1 on page 280
- Verifying the Route Advertisement to Device R4 on page 280
- Experimenting with the Default BGP Export Policy on page 280

Verifying the Routes on Device R1

Purpose On Device R1, check the static routes in the routing table.

Action

```
user@R1> show route protocol static

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.16/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.32/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.48/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.64/28     *[Static/5] 1d 02:02:13
                      Discard
```

Meaning Device R1 has four static routes.

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the static routes are advertised to Device R4.

Action

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
```

Meaning As expected, Device R1 only advertises its static routes to Device R4.

Experimenting with the Default BGP Export Policy

Purpose See what can happen when you remove the final **then reject** term from the policy **main** or the policy **subroutine**.

Action

1. On Device R1, deactivate the final term in the policy **main**.

```
[edit policy-options policy-statement main]
user@R1# deactivate term nothing-else
user@R1# commit
```

2. On Device R1, check to see which routes are advertised to Device R4.

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Now, all the BGP routes from Device R1 are sent to Device R4. This is because after the processing is returned to policy **main**, the default BGP export policy takes effect.

3. On Device R1, reactivate the final term in the policy **main**, and deactivate the final term in the policy **subroutine**.

```
[edit policy-options policy-statement main]
user@R1# activate term nothing-else


[edit policy-options policy-statement subroutine]
user@R1# deactivate term nothing-else
user@R1# commit
```

4. On Device R1, check to see which routes are advertised to Device R4.

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Now, all the BGP routes from Device R1 are sent to Device R4. This is because before the processing is returned to policy **main**, the default BGP export policy takes effect in the policy **subroutine**.

Meaning To prevent the default BGP export policy from taking effect, you must include a final **then reject** term in the main policy and in all referenced subroutines.

Related Documentation

- Understanding Policy Subroutines in Routing Policy Match Conditions on page 51
- How a Routing Policy Subroutine Is Evaluated on page 54

CHAPTER 20

AS Paths

- Example: Using AS Path Regular Expressions on page 283
- Example: Configuring a Routing Policy to Prepend the AS Path on page 292
- Example: Advertising Multiple Paths in BGP on page 301

Example: Using AS Path Regular Expressions

An autonomous system (AS) path is a route attribute used by BGP. The AS path is used both for route selection and to prevent potential routing loops. This example shows how to use regular expressions with AS path numbers to locate a set of routes.

- Requirements on page 283
- Overview on page 283
- Configuration on page 284
- Verification on page 290

Requirements

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

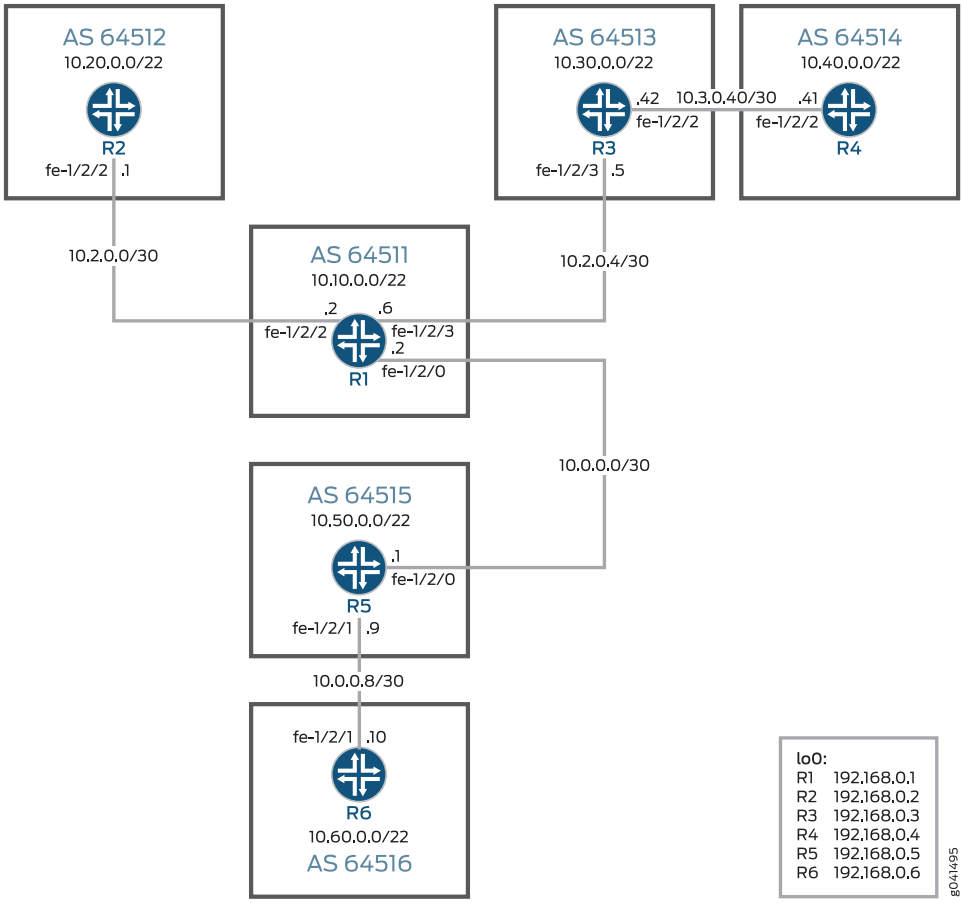
Overview

Figure 29 on page 284 shows several ASs connected through external BGP (EBGP) peering sessions. Each device is generating customer routes within its assigned address space.

Topology

Figure 29 on page 284 shows the sample network.

Figure 29: BGP Topology AS Regular Expressions



The administrators of AS 64516 want to reject all routes originating in AS 64513 and AS 64514. Two AS path regular expressions called **orig-in-64513** and **orig-in-64514** are created and referenced in a policy called **reject-some-routes**. The routing policy is then applied as an import policy on Device R6.

"CLI Quick Configuration" on page 284 shows the configuration for all of the devices in Figure 29 on page 284.

The section "Step-by-Step Procedure" on page 287 describes the steps on Device R2 and Device R6. "Verification" on page 290 shows how to use the **aspath-regex** option with the **show route** command on Device R2 to locate routes using regular expressions.

Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/2 unit 0 description to-R2
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.2/30
set interfaces fe-1/2/3 unit 0 description to-R3
```

```
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.6/30
set interfaces fe-1/2/0 unit 0 description to-R5
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp export send-static
set protocols bgp group 64512 type external
set protocols bgp group 64512 peer-as 64512
set protocols bgp group 64512 neighbor 10.2.0.1
set protocols bgp group 64513 type external
set protocols bgp group 64513 peer-as 64513
set protocols bgp group 64513 neighbor 10.2.0.5
set protocols bgp group 64515 type external
set protocols bgp group 64515 peer-as 64515
set protocols bgp group 64515 neighbor 10.0.0.1
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 10.10.1.0/24 reject
set routing-options static route 10.10.2.0/24 reject
set routing-options static route 10.10.3.0/24 reject
set routing-options autonomous-system 64511
```

Device R2

```
set interfaces fe-1/2/2 unit 0 description to-R1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.2.0.2
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 10.20.1.0/24 reject
set routing-options static route 10.20.2.0/24 reject
set routing-options static route 10.20.3.0/24 reject
set routing-options autonomous-system 64512
```

Device R3

```
set interfaces fe-1/2/3 unit 0 description to-R1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.5/30
set interfaces fe-1/2/2 unit 0 description to-R4
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.42/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.2.0.6
set protocols bgp group 64514 type external
set protocols bgp group 64514 peer-as 64514
set protocols bgp group 64514 neighbor 10.3.0.41
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 10.30.1.0/24 reject
set routing-options static route 10.30.2.0/24 reject
set routing-options static route 10.30.3.0/24 reject
set routing-options autonomous-system 64513
```

Device R4

```
set interfaces fe-1/2/2 unit 0 description to-R3
```

```
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.41/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp export send-static
set protocols bgp group 64513 type external
set protocols bgp group 64513 peer-as 64513
set protocols bgp group 64513 neighbor 10.3.0.42
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 10.40.1.0/24 reject
set routing-options static route 10.40.2.0/24 reject
set routing-options static route 10.40.3.0/24 reject
set routing-options autonomous-system 64514
```

Device R5

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 0 description to-R6
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.0.0.2
set protocols bgp group 64516 type external
set protocols bgp group 64516 peer-as 64516
set protocols bgp group 64516 neighbor 10.0.0.10
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 10.50.1.0/24 reject
set routing-options static route 10.50.2.0/24 reject
set routing-options static route 10.50.3.0/24 reject
set routing-options autonomous-system 64515
```

Device R6

```
set interfaces fe-1/2/1 unit 0 description to-R5
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols bgp export send-static
set protocols bgp group 64515 type external
set protocols bgp group 64515 import reject-some-routes
set protocols bgp group 64515 peer-as 64515
set protocols bgp group 64515 neighbor 10.0.0.9
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement reject-some-routes term find-routes from as-path
  orig-in-64513
set policy-options policy-statement reject-some-routes term find-routes from as-path
  orig-in-64514
set policy-options policy-statement reject-some-routes term find-routes then reject
set policy-options as-path orig-in-64513 ".* 64513"
set policy-options as-path orig-in-64514 ".* 64514"
set routing-options static route 10.60.1.0/24 reject
set routing-options static route 10.60.2.0/24 reject
set routing-options static route 10.60.3.0/24 reject
set routing-options autonomous-system 64516
```

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 R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/2 unit 0 description to-R1
user@R2# set fe-1/2/2 unit 0 family inet address 10.2.0.1/30

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]
user@R2# set export send-static
user@R2# set group 64511 type external
user@R2# set group 64511 peer-as 64511
user@R2# set group 64511 neighbor 10.2.0.2
```

3. Configure the routing policy.

```
[edit policy-options policy-statement send-static term 1]
user@R2# set from protocol static
user@R2# set then accept
```

4. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.20.1.0/24 reject
user@R2# set route 10.20.2.0/24 reject
user@R2# set route 10.20.3.0/24 reject
```

5. Configure the AS number.

```
[edit routing-options]
user@R2# set autonomous-system 64512
```

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 R6:

1. Configure the device interfaces.

```
[edit interfaces]
user@R6# set fe-1/2/1 unit 0 description to-R5
user@R6# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30

user@R6# set lo0 unit 0 family inet address 192.168.0.6/32
```

2. Configure the EBGP connection to Device R5.

```
[edit protocols bgp]
user@R6# set export send-static
user@R6# set group 64515 type external
```

```
user@R6# set group 64515 import reject-some-routes
user@R6# set group 64515 peer-as 64515
user@R6# set group 64515 neighbor 10.0.0.9
```

3. Configure the routing policy that sends static routes.

```
[edit policy-options policy-statement send-static term 1]
user@R6# set from protocol static
user@R6# set then accept
```

4. Configure the routing policy that rejects certain routes.

```
[edit policy-options policy-statement reject-some-routes term find-routes]
user@R6# set from as-path orig-in-64513
user@R6# set from as-path orig-in-64514
user@R6# set then reject


[edit policy-options]
user@R6# set as-path orig-in-64513 ".* 64513"
user@R6# set as-path orig-in-64514 ".* 64514"
```

5. Configure the static routes.

```
[edit routing-options static]
user@R6# set route 10.60.1.0/24 reject
user@R6# set route 10.60.2.0/24 reject
user@R6# set route 10.60.3.0/24 reject
```

6. Configure the AS number.

```
[edit routing-options]
user@R6# set autonomous-system 64516
```

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 R2

```
user@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        description to-R1;
        family inet {
            address 10.2.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}

user@R2# show protocols
bgp {
    export send-static;
```

```
    group 64511 {
      type external;
      peer-as 64511;
      neighbor 10.2.0.2;
    }
}
```

```
user@R2# show policy-options
policy-statement send-static {
    term 1 {
      from protocol static;
      then accept;
    }
}
```

```
user@R2# show routing-options
static {
    route 10.20.1.0/24 reject;
    route 10.20.2.0/24 reject;
    route 10.20.3.0/24 reject;
}
autonomous-system 64512;
```

Device R6

```
user@R6# show interfaces
fe-1/2/0 {
    unit 0 {
      description to-R5;
      family inet {
        address 10.0.0.10/30;
      }
    }
}
lo0 {
    unit 0 {
      family inet {
        address 192.168.0.6/32;
      }
    }
}
```

```
user@R6# show protocols
bgp {
    export send-static;
    group 64515 {
      type external;
      import reject-some-routes;
      peer-as 64515;
      neighbor 10.0.0.9;
    }
}
```

```
user@R6# show policy-options
policy-statement reject-some-routes {
    term find-routes {
      from as-path [ orig-in-64513 orig-in-64514 ];
      then reject;
    }
```

```
}
policy-statement send-static {
  term 1 {
    from protocol static;
    then accept;
  }
}
as-path orig-in-64513 ".* 64513";
as-path orig-in-64514 ".* 64514";
```

```
user@R6# show routing-options
static {
  route 10.60.1.0/24 reject;
  route 10.60.2.0/24 reject;
  route 10.60.3.0/24 reject;
}
autonomous-system 64516;
```

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

Verification

Confirm that the configuration is working properly.

- Finding Routes on Device R2 on page 290
- Making Sure That Routes Are Excluded on Device R6 on page 291

Finding Routes on Device R2

Purpose On Device R2, use the show route aspath-regex command to locate routes using regular expressions.

Action Look for routes that are originated by Device R6 in AS 64516.

```
user@R2> show route terse aspath-regex ".* 64516"

inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.60.1.0/24       B 170        100                           64511 64515
64516 I
  unverified                                      >10.2.0.2
* ? 10.60.2.0/24       B 170        100                           64511 64515
64516 I
  unverified                                      >10.2.0.2
* ? 10.60.3.0/24       B 170        100                           64511 64515
64516 I
  unverified                                      >10.2.0.2
```

Look for routes that are originated in either AS 64514 or AS 64516.

```
user@R2> show route terse aspath-regex ".* (64514|64516)"

inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
```

```
* ? 10.40.1.0/24       B 170        100                          64511 64513
64514 I
  unverified                                    >10.2.0.2
* ? 10.40.2.0/24       B 170        100                          64511 64513
64514 I
  unverified                                    >10.2.0.2
* ? 10.40.3.0/24       B 170        100                          64511 64513
64514 I
  unverified                                    >10.2.0.2
* ? 10.60.1.0/24       B 170        100                          64511 64515
64516 I
  unverified                                    >10.2.0.2
* ? 10.60.2.0/24       B 170        100                          64511 64515
64516 I
  unverified                                    >10.2.0.2
* ? 10.60.3.0/24       B 170        100                          64511 64515
64516 I
  unverified                                    >10.2.0.2
```

Look for routes that use AS 64513 as a transit network.

```
user@R2> show route terse aspath-regex ".* 64513 .+"

inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.40.1.0/24       B 170        100                          64511 64513
64514 I
  unverified                                    >10.2.0.2
* ? 10.40.2.0/24       B 170        100                          64511 64513
64514 I
  unverified                                    >10.2.0.2
* ? 10.40.3.0/24       B 170        100                          64511 64513
64514 I
  unverified
```

Meaning The output shows the routing table entries that match the specified AS path regular expressions.

Making Sure That Routes Are Excluded on Device R6

Purpose On Device R6, use the show route **and** show route hidden commands to make sure that routes originating from AS 64513 and AS 64514 are excluded from Device R6's routing table.

Action

```
user@R6> show route 10.30.0/22
inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)

user@R6> show route 10.40.0/22
inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)

user@R6> show route hidden

inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)
+ = Active Route, - = Last Active, * = Both

10.30.1.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.30.2.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.30.3.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.40.1.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
10.40.2.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
10.40.3.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
```

Meaning The output shows that the 10.30.0/22 and 10.40.0/22 routes are rejected on Device R6.

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 57
- Example: Testing a Routing Policy with Complex Regular Expressions on page 367

Example: Configuring a Routing Policy to Prepend the AS Path

This example shows how to configure a routing policy to prepend the AS path.

- Requirements on page 293
- Overview on page 293
- Configuration on page 294
- Verification on page 298

Requirements

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

Overview

The BGP best path algorithm determines how the best path to an autonomous system (AS) is selected.

In this example, an enterprise (AS 64510) has multihoming to two service providers, one in AS 64513 and the other in AS 64514. The enterprise prefers that incoming traffic take a particular path to reach its network because it has one fast primary connection and some slower connections that are to be used as backups if the primary connection is down. AS path prepending is used to influence inbound routing to the enterprise AS.

BGP does not take bandwidth into consideration when determining the best path. By using AS path prepending, you can lengthen the AS path that routing devices advertise to neighbors to make the neighbors calculate the path to be longer than it actually is.

AS 64510 has a T1 connection to AS 64513 and a T1 connection to AS 64514.

AS 64513 and AS 64514 also peer directly with each other, and have a 100-Mbps connection between them.

In addition, AS 64510 has a 100-Mbps connection from Device R2 to Device R3 in AS 64513.

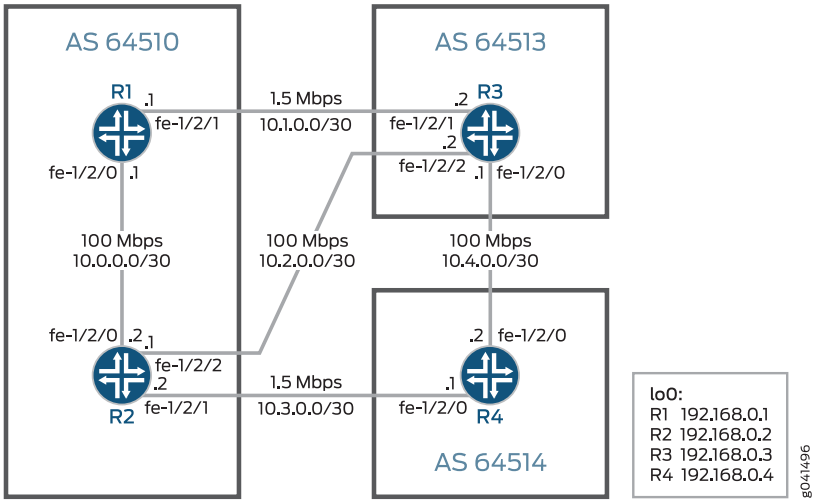
Because of the large difference in bandwidth, the administrators of AS 64510 prefer that all (or, at least, the majority) of incoming traffic come in over this much faster 100-Mbps Ethernet connection. AS 64510 is connected directly to Device R4, however, and this results in a short AS path in Device R4's routing table to AS 64510's prefixes. AS 64514, by default, sends any traffic for AS 64510 over its 1.5-Mbps T1 connection.

This example uses AS prepending to cause Device R4 to, instead, send any traffic destined to AS 64510 over the 100-Mbps connection to AS 64513. AS prepending is used to cause AS 64513 to send any traffic destined to AS 64510 over the 100-Mbps connection to Device R2 in AS 64510.

Topology

Figure 30 on page 294 shows the sample network.

Figure 30: BGP Topology AS Path Prepending



"CLI Quick Configuration" on page 294 shows the configuration for all of the devices in Figure 30 on page 294.

The section "Step-by-Step Procedure" on page 296 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.1.1/24
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int export send-routes
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64513
set protocols bgp group ext neighbor 10.1.0.2 export send-routes-and-prepend
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 send-routes term 1 from route-filter 172.16.0.0/16
    orlonger
set policy-options policy-statement send-routes term 1 from route-filter 10.0.0.0/8
    orlonger
set policy-options policy-statement send-routes term 1 then accept
set policy-options policy-statement send-routes-and-prepend term 1 from route-filter
    172.16.0.0/16 orlonger
set policy-options policy-statement send-routes-and-prepend term 1 then
    as-path-prepend "64510 64510"
set policy-options policy-statement send-routes-and-prepend term 1 then accept
set routing-options autonomous-system 64510
```

Device R2
```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family inet address 172.16.2.1/24
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int export send-routes
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group ext type external
set protocols bgp group ext neighbor 10.3.0.1 export send-routes-and-prepend
set protocols bgp group ext neighbor 10.3.0.1 peer-as 64514
set protocols bgp group ext neighbor 10.2.0.2 export send-routes
set protocols bgp group ext neighbor 10.2.0.2 peer-as 64513
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-routes term 1 from route-filter 10.0.0.0/8
  orlonger
set policy-options policy-statement send-routes term 1 from route-filter 172.16.0.0/16
  orlonger
set policy-options policy-statement send-routes term 1 then accept
set policy-options policy-statement send-routes-and-prepend term 1 from route-filter
  172.16.0.0/16 orlonger
set policy-options policy-statement send-routes-and-prepend term 1 then
  as-path-prepend "64510 64510"
set policy-options policy-statement send-routes-and-prepend term 1 then accept
set routing-options autonomous-system 64510
```

Device R3
```
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/0 unit 0 family inet address 10.4.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.3.1/24
set protocols bgp group ext type external
set protocols bgp group ext export send-routes
set protocols bgp group ext neighbor 10.2.0.1 peer-as 64510
set protocols bgp group ext neighbor 10.1.0.1 peer-as 64510
set protocols bgp group ext neighbor 10.4.0.2 peer-as 64514
set policy-options policy-statement send-routes term 1 from route-filter 172.16.0.0/16
  orlonger
set policy-options policy-statement send-routes term 1 then accept
set routing-options autonomous-system 64513
```

Device R4
```
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.1/30
set interfaces fe-1/2/0 unit 0 family inet address 10.4.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set interfaces lo0 unit 0 family inet address 172.16.4.1/24
set protocols bgp group ext type external
set protocols bgp group ext export send-routes
set protocols bgp group ext neighbor 10.3.0.2 peer-as 64510
set protocols bgp group ext neighbor 10.4.0.1 peer-as 64513
set policy-options policy-statement send-routes term 1 from route-filter 172.16.0.0/16
  orlonger
set policy-options policy-statement send-routes term 1 then accept
set routing-options autonomous-system 64514
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *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-1/2/2 unit 0 family inet address 10.2.0.1/30

user@R2# set fe-1/2/1 unit 0 family inet address 10.3.0.2/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
user@R2# set lo0 unit 0 family inet address 172.16.2.1/24
```

2. Configure an interior gateway protocol (IGP) connection to Device R1.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.0 passive
user@R2# set interface fe-1/2/0.0
```

3. Create the routing policy that will be applied to Device R1 and Device R3.

   This policy simply advertises the routes that are in Device R2's routing table.

```
[edit policy-options policy-statement send-routes term 1]
user@R2# set from route-filter 10.0.0.0/8 orlonger
user@R2# set from route-filter 172.16.0.0/16 orlonger
user@R2# set then accept
```

4. Create the routing policy that will be applied to Device R4.

   This policy prepends the routes that are advertised to Device R4, so that Device R4 is less likely to use its direct connection to Device R2.

NOTE: If you enter multiple numbers, you must separate each number with a space. Enclose the numbers in double quotation marks.

```
[edit policy-options policy-statement send-routes-and-prepend term 1]
user@R2# set from route-filter 172.16.0.0/16 orlonger
user@R2# set then as-path-prepend "64510 64510"
user@R2# set then accept
```

5. Configure the internal BGP (IBGP) connection to Device R1.

```
[edit protocols bgp group int]
user@R2# set type internal
user@R2# set local-address 192.168.0.2
user@R2# set export send-routes
user@R2# set neighbor 192.168.0.1
```

6. Configure the external BGP (EBGP) connection to Device R3 and Device R4.

   For Device R3, apply the **send-routes** policy. For Device R4, apply the **send-routes-and-prepend** policy.

   ```
   [edit protocols bgp group ext]
   user@R2# set type external
   user@R2# set neighbor 10.3.0.1 export send-routes-and-prepend
   user@R2# set neighbor 10.3.0.1 peer-as 64514
   user@R2# set neighbor 10.2.0.2 export send-routes
   user@R2# set neighbor 10.2.0.2 peer-as 64513
   ```

7. Configure the AS number.

   ```
   [edit routing-options]
   user@R2# set autonomous-system 64510
   ```

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 10.3.0.2/30;
    }
  }
}
fe-1/2/2 {
  unit 0 {
    family inet {
      address 10.2.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.2/32;
      address 172.16.2.1/24;
    }
  }
}
```

```
user@R2# show protocols
bgp {
  group int {
```

```
        type internal;
        local-address 192.168.0.2;
        export send-routes;
        neighbor 192.168.0.1;
      }
      group ext {
        type external;
        neighbor 10.3.0.1 {
          export send-routes-and-prepend;
          peer-as 64514;
        }
        neighbor 10.2.0.2 {
          export send-routes;
          peer-as 64513;
        }
      }
    }
    ospf {
      area 0.0.0.0 {
        interface lo0.0 {
          passive;
        }
        interface fe-1/2/0.0;
      }
    }

    user@R2# show policy-options
    policy-statement send-routes {
      term 1 {
        from {
          route-filter 10.0.0.0/8 orlonger;
          route-filter 172.16.0.0/16 orlonger;
        }
        then accept;
      }
    }
    policy-statement send-routes-and-prepend {
      term 1 {
        from {
          route-filter 172.16.0.0/16 orlonger;
        }
        then {
          as-path-prepend "64510 64510";
          accept;
        }
      }
    }

    user@R2# show routing-options
    autonomous-system 64510;
```

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

Verification

Confirm that the configuration is working properly.

Verifying That the Routes Are Prepended

Purpose On Device R3 and Device R4, use the **show route** command to view the routing tables.

Action

```
user@R3> show route match-prefix 172.16.*/24 protocol bgp

inet.0: 17 destinations, 30 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 03:36:53, MED 1, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.1 via fe-1/2/2.0
                    [BGP/170] 03:37:08, localpref 100
                      AS path: 64510 64510 64510 I, validation-state: unverified

                    > to 10.1.0.1 via fe-1/2/1.0
172.16.2.0/24      *[BGP/170] 03:37:07, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.1 via fe-1/2/2.0
                    [BGP/170] 03:36:53, MED 1, localpref 100
                      AS path: 64510 64510 64510 I, validation-state: unverified

                    > to 10.1.0.1 via fe-1/2/1.0
172.16.4.0/24      *[BGP/170] 03:37:31, localpref 100
                      AS path: 64514 I, validation-state: unverified
                    > to 10.4.0.2 via fe-1/2/0.0
                    [BGP/170] 03:37:07, localpref 100
                      AS path: 64510 64514 I, validation-state: unverified
                    > to 10.2.0.1 via fe-1/2/2.0
                    [BGP/170] 03:36:35, localpref 100
                      AS path: 64510 64510 64510 64514 I, validation-state:
unverified
                    > to 10.1.0.1 via fe-1/2/1.0

user@R4> show route match-prefix 172.16.* protocol bgp

inet.0: 15 destinations, 22 routes (15 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 03:34:31, localpref 100
                      AS path: 64513 64510 I, validation-state: unverified
                    > to 10.4.0.1 via fe-1/2/0.0
                    [BGP/170] 03:34:31, MED 1, localpref 100
                      AS path: 64510 64510 64510 I, validation-state: unverified

                    > to 10.3.0.2 via fe-1/2/1.0
172.16.2.0/24      *[BGP/170] 03:34:45, localpref 100
                      AS path: 64513 64510 I, validation-state: unverified
                    > to 10.4.0.1 via fe-1/2/0.0
                    [BGP/170] 03:35:09, localpref 100
                      AS path: 64510 64510 64510 I, validation-state: unverified

                    > to 10.3.0.2 via fe-1/2/1.0
172.16.3.0/24      *[BGP/170] 03:35:09, localpref 100
                      AS path: 64513 I, validation-state: unverified
                    > to 10.4.0.1 via fe-1/2/0.0
                    [BGP/170] 03:34:45, localpref 100
                      AS path: 64510 64510 64510 64513 I, validation-state:
unverified
                    > to 10.3.0.2 via fe-1/2/1.0
```

Meaning Active routes are marked with an asterisk (*). The output shows the AS paths that are prepended with “64510 64510” to make the AS path longer, and thereby make the

associated route inactive.

Device R3 does not use its fe-1/2/1.0 interface to reach 172.16 destinations in AS 64510. Instead, Device R3 uses its fe-1/2/2.0 interface, even for destinations that are advertised by Device R1.

Device R4 does not use its fe-1/2/1.0 interface to reach 172.16 destinations in AS 64510. Instead, Device R4 goes through Device R3 to reach AS 64510. This way, the higher bandwidth (fast) link is utilized, and the slower links are used as backups when the fast link is unavailable.

Related Documentation

- Understanding Prepending AS Numbers to BGP AS Paths on page 63
- Understanding Adding AS Numbers to BGP AS Paths on page 64
- *Understanding BGP Path Selection*

Example: Advertising Multiple Paths in BGP

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 Internet draft draft-ietf-idr-add-paths-04, *Advertisement of Multiple Paths in BGP*.

- Requirements on page 301
- Overview on page 301
- Configuration on page 302
- Verification on page 321

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 3D Universal Edge Routers, or T Series Core Routers.
- The three routers must be running Junos OS Release 11.4 or later.

Overview

The following statements are used for configuring multiple paths to a destination:

```
[edit protocols bgp group group-name family family]
add-path {
   receive;
   send {
```

```
        path-count number;
        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.

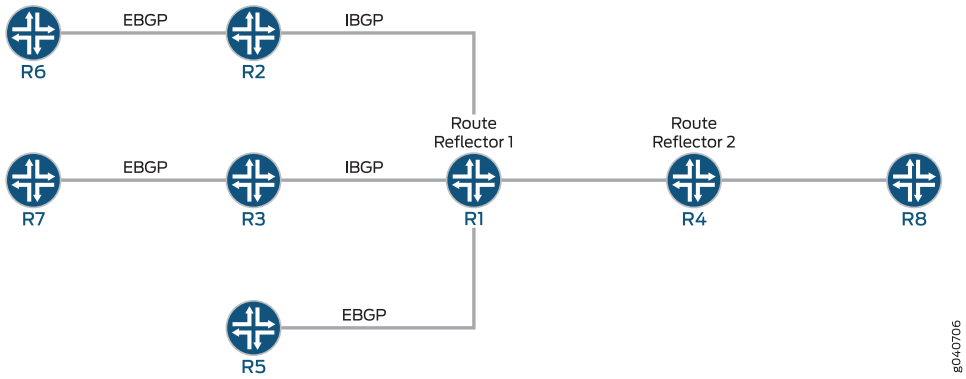
With the **add-path receive** configuration, Router R8 is configured to receive multiple paths from Router R4.

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 199.1.1.1/32 route.

Topology Diagram

Figure 31 on page 302 shows the topology used in this example.

Figure 31: Advertisement of Multiple Paths in BGP



Configuration

- Configuring Router R1 on page 305
- Configuring Router R2 on page 308
- Configuring Router R3 on page 310
- Configuring Router R4 on page 312
- Configuring Router R5 on page 314

- Configuring Router R6 on page 316
- Configuring Router R7 on page 317
- Configuring Router R8 on page 319
- Results on page 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.

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 routing-options autonomous-system 1
set policy-options policy-statement allow_199 from route-filter 199.1.1.1/32 exact
set policy-options policy-statement allow_199 then accept
```

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 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 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 199.1.1.1/32 reject
set routing-options static route 198.1.1.1/32 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 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 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 routing-options autonomous-system 2
set routing-options static route 199.1.1.1/32 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 *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 199.1.1.1/32 route.

   Router R4 receives multiple paths for the 198.1.1.1/32 route and the 199.1.1.1/32 route. However, because of this policy, Router R4 only sends multiple paths for the 199.1.1.1/32 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 199.1.1.1/32 exact
user@R4# set then accept
```

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 199.1.1.1/32 exact;
      }
      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 199.1.1.1/32 reject
user@R5# set static route 198.1.1.1/32 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 198.1.1.1/32 reject;
   route 199.1.1.1/32 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 199.1.1.1/32 reject
   user@R6# set routing-options static route 198.1.1.1/32 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 198.1.1.1/32 reject;
    route 199.1.1.1/32 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 199.1.1.1/32 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 199.1.1.1/32 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

Confirm that the configuration is working properly.

- Verifying That the BGP Peers Have the Ability to Send and Receive Multiple Paths on page 321
- Verifying That Router R1 Is Advertising Multiple Paths on page 321
- Verifying That Router R4 Is Receiving and Advertising Multiple Paths on page 322
- Verifying That Router R8 Is Receiving Multiple Paths on page 323
- Checking the Path ID on page 323

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+65237 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+65237 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 198.1.1.1/32 destination and multiple paths to the 199.1.1.1/32 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
* 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
* 200.1.1.0/30            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 199.1.1.1/32 destination are received from Router R1 and advertised to Router R8. Make sure that multiple paths to the 198.1.1.1/32 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
* 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
* 200.1.1.0/30            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
* 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
                          10.0.15.2                    100        2 2 I
* 200.1.1.0/30            10.0.0.20                    100        2 I
```

Meaning The **show route receive-protocol** command shows that Router R4 receives two paths to the 198.1.1.1/32 destination and three paths to the 199.1.1.1/32 destination. The **show route advertising-protocol** command shows that Router R4 advertises only one path to the 198.1.1.1/32 destination and advertises all three paths to the 199.1.1.1/32 destination.

Because of the prefix policy that is applied to Router R4, Router R4 does not advertise multiple paths to the 198.1.1.1/32 destination. Router R4 advertises only one path to the 198.1.1.1/32 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 199.1.1.1/32 destination through Router R4. Make sure that Router R8 receives only one path to the 198.1.1.1/32 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
* 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
                          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 199.1.1.1/32 detail**

```
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
199.1.1.1/32 (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+65237
                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+65237
                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+65237
                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 199.1.1.1/32 detail

inet.0: 17 destinations, 19 routes (17 active, 0 holddown, 0 hidden)
199.1.1.1/32 (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
```

Related Documentation

- *Understanding the Advertisement of Multiple Paths to a Single Destination in BGP*
- Understanding Adding AS Numbers to BGP AS Paths on page 64

CHAPTER 21

Communities

- Example: Configuring Communities in a Routing Policy on page 327
- Example: Configuring Extended Communities in a Routing Policy on page 342
- Example: Defining a Routing Policy Based on the Number of BGP Communities on page 351
- Example: Defining a Routing Policy That Removes BGP Communities on page 358

Example: Configuring Communities in a Routing Policy

A community is a route attribute used by BGP to administratively group routes with similar properties.

- Requirements on page 327
- Overview on page 327
- Configuration on page 329
- Verification on page 337

Requirements

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

Overview

One main role of the community attribute is to be an administrative tag value used to associate routes together. Generally, these routes share some common properties, but that is not required. Communities are a flexible tool within BGP. An individual community value can be assigned to a single route or multiple routes. A route can be assigned a single community value or multiple values. Networks use the community attribute to assist in implementing administrative routing policies. A route's assigned value can allow it to be accepted into the network, or rejected from the network, or allow it to modify attributes.

Figure 32 on page 329 shows Device R1, Device R2, and Device R3 as internal BGP (IBGP) peers in autonomous system (AS) 64510. Device R4 is advertising the 172.16.0.0/21

address space from AS 64511. The specific routes received by Device R1 from Device R4 are as follows:

```
user@R1> show route receive-protocol bgp 10.0.0.13
inet.0: 20 destinations, 28 routes (20 active, 0 holddown, 8 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.0.0/24           10.0.0.13                               64511 I
* 172.16.1.0/24           10.0.0.13                               64511 I
* 172.16.2.0/24           10.0.0.13                               64511 I
* 172.16.3.0/24           10.0.0.13                               64511 I
* 172.16.4.0/24           10.0.0.13                               64511 I
* 172.16.5.0/24           10.0.0.13                               64511 I
* 172.16.6.0/24           10.0.0.13                               64511 I
* 172.16.7.0/24           10.0.0.13                               64511 I
```

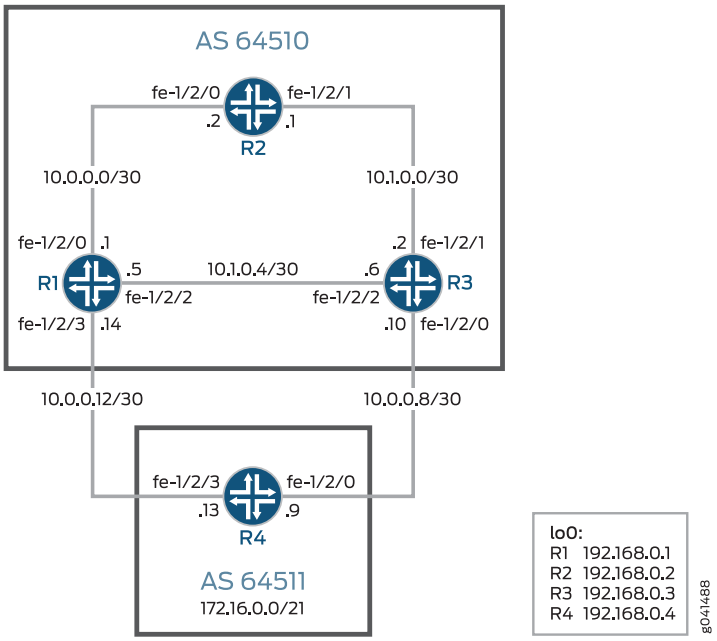
The administrators of AS 64511 want to receive certain user traffic from Device R1, and other user traffic from Device R3. To accomplish this administrative goal, Device R4 attaches the community value of 64511:1 to some routes that it sends and attaches the community value 64511:3 to other routes that it sends. Routing policies within AS 64510 are configured using a community match criterion to change the local preference of the received routes to new values that alter the BGP route selection algorithm. The route with the highest local preference value is preferred.

On Device R1, routes with the 64511:1 community value are assigned a local preference of 200, and routes with the 64511:3 community value are assigned a local preference of 50. On Device R3, the reverse is done so that routes with the 64511:3 community value are assigned a local preference of 200, and routes with the 64511:1 community value are assigned a local preference of 50. This information is then communicated through IBGP by both Device R1 and Device R3 to Device R2.

Topology

Figure 32 on page 329 shows the sample network.

Figure 32: Topology for Regular BGP Communities



"CLI Quick Configuration" on page 329 shows the configuration for all of the devices in Figure 32 on page 329.

The section "Step-by-Step Procedure" on page 332 describes the steps on Device R1 and R4.

Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
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 export send-direct
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group ext type external
set protocols bgp group ext import change-local-preference
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.13
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement change-local-preference term find-R1-routes from
    community from-R1
set policy-options policy-statement change-local-preference term find-R1-routes then
    local-preference 200
```

```
set policy-options policy-statement change-local-preference term find-R3-routes from
    community from-R3
set policy-options policy-statement change-local-preference term find-R3-routes then
    local-preference 50
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.0.0.12/30
    exact
set policy-options policy-statement send-direct term 1 then accept
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

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 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 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 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 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 8 family inet address 10.1.0.6/30
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.10/30
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 neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2 export send-direct
set protocols bgp group ext type external
set protocols bgp group ext import change-local-preference
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.9
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.8
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement change-local-preference term find-R3-routes from
    community from-R3
set policy-options policy-statement change-local-preference term find-R3-routes then
    local-preference 200
set policy-options policy-statement change-local-preference term find-R1-routes from
    community from-R1
set policy-options policy-statement change-local-preference term find-R1-routes then
    local-preference 50
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.0.0.8/30 exact
set policy-options policy-statement send-direct term 1 then accept
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options router-id 192.168.0.3
```

```
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.9/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group to-R1 type external
set protocols bgp group to-R1 export send-static
set protocols bgp group to-R1 peer-as 64510
set protocols bgp group to-R1 neighbor 10.0.0.14
set protocols bgp group to-R3 type external
set protocols bgp group to-R3 export send-static
set protocols bgp group to-R3 peer-as 64510
set protocols bgp group to-R3 neighbor 10.0.0.10
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 from route-filter 172.16.0.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.1.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.2.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.3.0/24
  exact
set policy-options policy-statement send-static term 1 then community add from-R1
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 from protocol static
set policy-options policy-statement send-static term 2 from route-filter 172.16.4.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.5.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.6.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.7.0/24
  exact
set policy-options policy-statement send-static term 2 then community add from-R3
set policy-options policy-statement send-static term 2 then accept
set policy-options policy-statement send-static term 3 then reject
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options static route 172.16.0.0/24 reject
set routing-options static route 172.16.1.0/24 reject
set routing-options static route 172.16.2.0/24 reject
set routing-options static route 172.16.3.0/24 reject
set routing-options static route 172.16.4.0/24 reject
set routing-options static route 172.16.5.0/24 reject
set routing-options static route 172.16.6.0/24 reject
set routing-options static route 172.16.7.0/24 reject
set routing-options router-id 192.168.0.4
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set fe-1/2/2 unit 0 family inet address 10.1.0.5/30

user@R1# set fe-1/2/3 unit 0 family inet address 10.0.0.14/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure internal gateway protocol (IGP) connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

3. Configure the IBGP connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2 export send-direct
user@R1# set neighbor 192.168.0.3
```

4. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set import change-local-preference
user@R1# set peer-as 64511
user@R1# set neighbor 10.0.0.13
```

5. Configure the policy **send-direct**.

   This policy is referenced in the IBGP connection to Device R2 and enables Device R2 to have external reachability. An alternative is to configure a **next-hop self** policy on Device R1 and Device R3.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set from route-filter 10.0.0.12/30 exact
user@R1# set then accept
```

6. Configure the policy that changes the local preference for routes with specified community tags.

```
[edit policy-options policy-statement change-local-preference]
user@R1# set term find-R1-routes from community from-R1
user@R1# set term find-R1-routes then local-preference 200
user@R1# set term find-R3-routes from community from-R3
```

```
user@R1# set term find-R3-routes then local-preference 50

[edit policy-options ]
user@R1# set community from-R1 members 64511:1
user@R1# set community from-R3 members 64511:3
```

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

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

Step-by-Step Procedure

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

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces]
user@R4# set fe-1/2/0 unit 0 family inet address 10.0.0.9/30

user@R4# set fe-1/2/3 unit 0 family inet address 10.0.0.13/30

user@R4# set lo0 unit 0 family inet address 192.168.0.4/32
```

2. Configure the EBGP connection to Device R1 and Device R3.

```
[edit protocols bgp]
user@R4# set group to-R1 type external
user@R4# set group to-R1 export send-static
user@R4# set group to-R1 peer-as 64510
user@R4# set group to-R1 neighbor 10.0.0.14

user@R4# set group to-R3 type external
user@R4# set group to-R3 export send-static
user@R4# set group to-R3 peer-as 64510
user@R4# set group to-R3 neighbor 10.0.0.10
```

3. Configure the community tags.

```
[edit policy-options ]
user@R4# set community from-R1 members 64511:1
user@R4# set community from-R3 members 64511:3
```

4. Configure the policy **send-static**.

   This policy is referenced in the EBGP connections to Device R1 and Device R3. The policy attaches the 64511:1 (from-R1) community to some routes and the 64511:3 (from-R3) community to other routes.

```
[edit policy-options policy-statement send-static term 1]
user@R4# set from protocol static
user@R4# set from route-filter 172.16.0.0/24 exact
user@R4# set from route-filter 172.16.1.0/24 exact
user@R4# set from route-filter 172.16.2.0/24 exact
```

```
user@R4# set from route-filter 172.16.3.0/24 exact
user@R4# set then community add from-R1
user@R4# set then accept

[edit policy-options policy-statement send-static term 2]
user@R4# set from protocol static
user@R4# set from route-filter 172.16.4.0/24 exact
user@R4# set from route-filter 172.16.5.0/24 exact
user@R4# set from route-filter 172.16.6.0/24 exact
user@R4# set from route-filter 172.16.7.0/24 exact
user@R4# set then community add from-R3
user@R4# set then accept

[edit policy-options policy-statement send-static term 3]
user@R4# set then reject
```

5. Configure the static routes.

```
[edit routing-options static]
user@R4# set route 172.16.0.0/24 reject
user@R4# set route 172.16.1.0/24 reject
user@R4# set route 172.16.2.0/24 reject
user@R4# set route 172.16.3.0/24 reject
user@R4# set route 172.16.4.0/24 reject
user@R4# set route 172.16.5.0/24 reject
user@R4# set route 172.16.6.0/24 reject
user@R4# set route 172.16.7.0/24 reject
```

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

```
[edit routing-options]
user@R4# set router-id 192.168.0.4
user@R4# 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
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        family inet {
            address 10.1.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
```



```
      family inet {
        address 10.0.0.14/30;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.0.1/32;
      }
    }
  }

  user@R1# show protocols
  bgp {
    group int {
      type internal;
      local-address 192.168.0.1;
      neighbor 192.168.0.2 {
        export send-direct;
      }
      neighbor 192.168.0.3;
    }
    group ext {
      type external;
      import change-local-preference;
      peer-as 64511;
      neighbor 10.0.0.13;
    }
  }
  ospf {
    area 0.0.0.0 {
      interface fe-1/2/0.0;
      interface fe-1/2/2.0;
      interface lo0.0 {
        passive;
      }
    }
  }

  user@R1# show policy-options
  policy-statement change-local-preference {
    term find-R1-routes {
      from community from-R1;
      then {
        local-preference 200;
      }
    }
    term find-R3-routes {
      from community from-R3;
      then {
        local-preference 50;
      }
    }
  }
  policy-statement send-direct {
```

```
    term 1 {
      from {
        protocol direct;
        route-filter 10.0.0.12/30 exact;
      }
      then accept;
    }
  }
  community from-R1 members 64511:1;
  community from-R3 members 64511:3;

  user@R1# show routing-options
  router-id 192.168.0.1;
  autonomous-system 64510;
```

Device R4

```
  user@R4# show interfaces
  fe-1/2/0 {
    unit 0 {
      family inet {
        address 10.0.0.9/30;
      }
    }
  }
  fe-1/2/3 {
    unit 0 {
      family inet {
        address 10.0.0.13/30;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.0.4/32;
      }
    }
  }

  user@R4# show protocols
  bgp {
    group to-R1 {
      type external;
      export send-static;
      peer-as 64510;
      neighbor 10.0.0.14;
    }
    group to-R3 {
      type external;
      export send-static;
      peer-as 64510;
      neighbor 10.0.0.10;
    }
  }

  user@R4# show policy-options
  policy-statement send-static {
    term 1 {
```

```
        from {
            protocol static;
            route-filter 172.16.0.0/24 exact;
            route-filter 172.16.1.0/24 exact;
            route-filter 172.16.2.0/24 exact;
            route-filter 172.16.3.0/24 exact;
        }
        then {
            community add from-R1;
            accept;
        }
    }
    term 2 {
        from {
            protocol static;
            route-filter 172.16.4.0/24 exact;
            route-filter 172.16.5.0/24 exact;
            route-filter 172.16.6.0/24 exact;
            route-filter 172.16.7.0/24 exact;
        }
        then {
            community add from-R3;
            accept;
        }
    }
    term 3 {
        then reject;
    }
}
community from-R1 members 64511:1;
community from-R3 members 64511:3;
```

```
user@R4# show routing-options
static {
    route 172.16.0.0/24 reject;
    route 172.16.1.0/24 reject;
    route 172.16.2.0/24 reject;
    route 172.16.3.0/24 reject;
    route 172.16.4.0/24 reject;
    route 172.16.5.0/24 reject;
    route 172.16.6.0/24 reject;
    route 172.16.7.0/24 reject;
}
router-id 192.168.0.4;
autonomous-system 64511;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes Sent on Device R4 on page 338
- Verifying the Routes Received on Device R2 on page 340

Verifying the Routes Sent on Device R4

Purpose On Device R4, check the routes sent to Device R1 and Device R3.

Action `user@R4>` **show route advertising-protocol bgp 10.0.0.14**

```
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
* 172.16.0.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.1.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.2.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.3.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.4.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.5.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.6.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.7.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3
```

`user@R2>` **show route advertising-protocol bgp 10.0.0.10**

```
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
* 172.16.0.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1
```

```
* 172.16.1.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.2.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.3.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.4.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.5.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.6.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.7.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3
```

Meaning Device R4 has tagged the routes with the communities 64511:1 and 64511:3 and sent them to Device R1 and R3.

Verifying the Routes Received on Device R2

Purpose On Device R2, check the routes received from Device R1 and Device R3.

Action

```
user@R2> show route receive-protocol bgp 192.168.0.1

inet.0: 22 destinations, 30 routes (22 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.12/30            192.168.0.1                  100        I
* 172.16.0.0/24           10.0.0.13                    200        64511 I
* 172.16.1.0/24           10.0.0.13                    200        64511 I
* 172.16.2.0/24           10.0.0.13                    200        64511 I
* 172.16.3.0/24           10.0.0.13                    200        64511 I
  172.16.4.0/24           10.0.0.13                    50         64511 I
  172.16.5.0/24           10.0.0.13                    50         64511 I
  172.16.6.0/24           10.0.0.13                    50         64511 I
  172.16.7.0/24           10.0.0.13                    50         64511 I

user@R2> show route match-prefix 172.16.*

inet.0: 22 destinations, 30 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.1.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.2.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.3.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.4.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
172.16.5.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
```

```
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
172.16.6.0/24       *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
172.16.7.0/24       *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
                     [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
```

Meaning Device R2 has the routes with the expected local preferences and the expected active routes, as designated by the asterisks (*).

Related Documentation

- Example: Configuring Extended Communities in a Routing Policy on page 342
- Example: Defining a Routing Policy That Removes BGP Communities on page 358
- Example: Defining a Routing Policy Based on the Number of BGP Communities on page 351
- *Example: Redistributing BGP Routes with a Specific Community Tag into IS-IS*

Example: Configuring Extended Communities in a Routing Policy

An extended community is similar in most ways to a regular community. Some networking implementations, such as virtual private networks (VPNs), use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an eight-octet value divided into two main sections.

- Requirements on page 342
- Overview on page 343
- Configuration on page 343
- Verification on page 347

Requirements

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

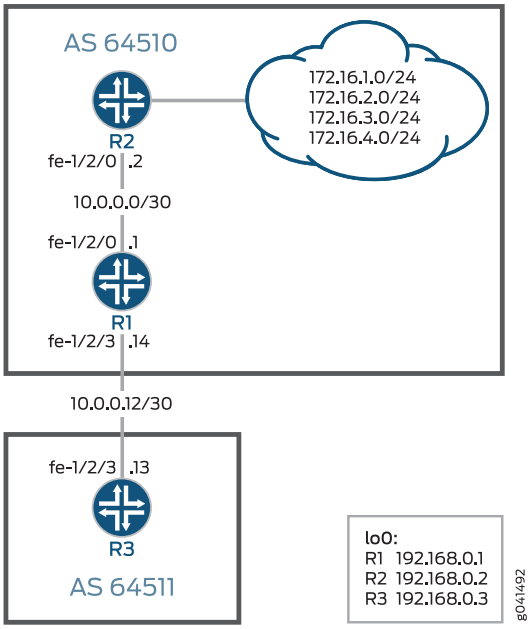
Overview

In this example, Device R1 and Device R2 are OSPF neighbors in autonomous system (AS) 64510. Device R3 has an external BGP (EBGP) connection to Device R1. Device R2 has customer networks in the 172.16/16 address space, simulated with addresses on its loopback interface (lo0). Device R1 has static routes to several 172.16.*x*/24 networks, and attaches regular community values to these routes. Device R1 then uses an export policy to advertise the routes to Device R3. Device R3 receives these routes and uses an import policy to add extended community values to the routes.

Topology

Figure 33 on page 343 shows the sample network.

Figure 33: Topology for Extended BGP Communities



"CLI Quick Configuration" on page 343 shows the configuration for all of the devices in Figure 33 on page 343.

The section "Step-by-Step Procedure" on page 345 describes the steps on Device R3.

Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32 primary
set protocols bgp group ext type external
set protocols bgp group ext export send-static
```

```
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.13
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 send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.1.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.1.0/24 community 64510:1
set routing-options static route 172.16.2.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.2.0/24 community 64510:2
set routing-options static route 172.16.3.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.3.0/24 community 64510:3
set routing-options static route 172.16.4.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.4.0/24 community 64510:4
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set interfaces lo0 unit 0 family inet address 172.16.3.3/32
set interfaces lo0 unit 0 family inet address 172.16.4.4/32
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group to-R1 type external
set protocols bgp group to-R1 import set-ext-comms
set protocols bgp group to-R1 peer-as 64510
set protocols bgp group to-R1 neighbor 10.0.0.14
set policy-options policy-statement set-ext-comms term route-1 from route-filter
  172.16.1.0/24 exact
set policy-options policy-statement set-ext-comms term route-1 then community add
  target-as
set policy-options policy-statement set-ext-comms term route-1 then accept
set policy-options policy-statement set-ext-comms term route-2 from route-filter
  172.16.2.0/24 exact
set policy-options policy-statement set-ext-comms term route-2 then community add
  target-ip
set policy-options policy-statement set-ext-comms term route-2 then accept
set policy-options policy-statement set-ext-comms term route-3 from route-filter
  172.16.3.0/24 exact
set policy-options policy-statement set-ext-comms term route-3 then community add
  origin-as
set policy-options policy-statement set-ext-comms term route-3 then accept
set policy-options policy-statement set-ext-comms term route-4 from route-filter
  172.16.4.0/24 exact
set policy-options policy-statement set-ext-comms term route-4 then community add
  origin-ip
set policy-options policy-statement set-ext-comms term route-4 then accept
set policy-options community origin-as members origin:64511:3
```

```
set policy-options community origin-ip members origin:172.16.7.7:4
set policy-options community target-as members target:64511:1
set policy-options community target-ip members target:172.16.7.7:2
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64511
```

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 R3:

1. Configure the interfaces.

```
[edit interfaces]
user@R3# set fe-1/2/3 unit 0 family inet address 10.0.0.13/30

user@R3# set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group to-R1]
user@R3# set type external
user@R3# set import set-ext-comms
user@R3# set peer-as 64510
user@R3# set neighbor 10.0.0.14
```

3. Configure the policy that adds extended community values to the routes received from Device R1.

   An extended community uses a notation of ***type*:*administrator*:*assigned-number***.

   The specific community values can be anything that accomplishes your administrative goals, within certain parameters, as explained in community.

```
[edit policy-options policy-statement set-ext-comms]
user@R3# set term route-1 from route-filter 172.16.1.0/24 exact
user@R3# set term route-1 then community add target-as
user@R3# set term route-1 then accept

user@R3# set term route-2 from route-filter 172.16.2.0/24 exact
user@R3# set term route-2 then community add target-ip
user@R3# set term route-2 then accept

user@R3# set term route-3 from route-filter 172.16.3.0/24 exact
user@R3# set term route-3 then community add origin-as
user@R3# set term route-3 then accept

user@R3# set term route-4 from route-filter 172.16.4.0/24 exact
user@R3# set term route-4 then community add origin-ip
user@R3# set term route-4 then accept

[edit policy-options]
user@R3# set community origin-as members origin:64511:3
user@R3# set community origin-ip members origin:172.16.7.7:4
user@R3# set community target-as members target:64511:1
```

```
user@R3# set community target-ip members target:172.16.7.7:2
```

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

```
[edit routing-options]
user@R3# set router-id 192.168.0.3
user@R3# 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.

```
user@R3# show interfaces
fe-1/2/3 {
    unit 0 {
        family inet {
            address 10.0.0.13/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.3/32;
        }
    }
}
```

```
user@R3# show protocols
bgp {
    group to-R1 {
        type external;
        import set-ext-comms;
        peer-as 64510;
        neighbor 10.0.0.14;
    }
}
```

```
user@R3# show policy-options
policy-statement set-ext-comms {
    term route-1 {
        from {
            route-filter 172.16.1.0/24 exact;
        }
        then {
            community add target-as;
            accept;
        }
    }
    term route-2 {
        from {
            route-filter 172.16.2.0/24 exact;
        }
        then {
            community add target-ip;
```



```
        accept;
      }
    }
    term route-3 {
      from {
        route-filter 172.16.3.0/24 exact;
      }
      then {
        community add origin-as;
        accept;
      }
    }
    term route-4 {
      from {
        route-filter 172.16.4.0/24 exact;
      }
      then {
        community add origin-ip;
        accept;
      }
    }
  }
  community origin-as members origin:64511:3;
  community origin-ip members origin:172.16.7.7:4;
  community target-as members target:64511:1;
  community target-ip members target:172.16.7.7:2;

  user@R3# show routing-options
  router-id 192.168.0.3;
  autonomous-system 64511;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device R1 on page 347
- Verifying the Routes on Device R3 on page 349

Verifying the Routes on Device R1

Purpose On Device R1, check the 172.16. routes in the routing table.

Action `user@R1>` **show route protocol static match-prefix 172.16.* detail**

```
inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:1

172.16.2.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:2

172.16.3.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:3

172.16.4.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:4
```

Meaning The output shows that the regular community values are attached to the routes.

NOTE: The communities are attached to static routes, thus demonstrating that communities can be attached to non-BGP routes.

Verifying the Routes on Device R3

Purpose On Device R3, check the 172.16. routes in the routing table.

Action

```
user@R3> show route protocol bgp match-prefix 172.16.* detail
betsy@tp5# run show route protocol bgp match-prefix 172.16.* detail logical-system
 R3

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:1 target:64511:1
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.2.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:2 target:172.16.7.7:2
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.3.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:3 origin:64511:3
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.4.0/24 (1 entry, 1 announced)
```

```
*BGP    Preference: 170/-101
        Next hop type: Router, Next hop index: 611
        Address: 0x9260130
        Next-hop reference count: 8
        Source: 10.0.0.14
        Next hop: 10.0.0.14 via fe-1/2/3.0, selected
        State: <Active Ext>
        Local AS: 64511 Peer AS: 64510
        Age: 1:57:27
        Task: BGP_64510.10.0.0.14+54618
        Announcement bits (1): 0-KRT
        AS path: 64510 I
        Communities: 64510:4 origin:172.16.7.7:4
        Accepted
        Localpref: 100
        Router ID: 192.168.0.1
```

Meaning The output shows that the regular community values remain attached to the routes, and the extended community values are added.

Related Documentation

- Example: Configuring Communities in a Routing Policy on page 327
- Example: Defining a Routing Policy That Removes BGP Communities on page 358
- Example: Defining a Routing Policy Based on the Number of BGP Communities on page 351
- *Example: Redistributing BGP Routes with a Specific Community Tag into IS-IS*

Example: Defining a Routing Policy Based on the Number of BGP Communities

This example shows how to create a policy that accepts BGP routes based on the number of BGP communities.

- Requirements on page 351
- Overview on page 351
- Configuration on page 352
- Verification on page 356

Requirements

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

Overview

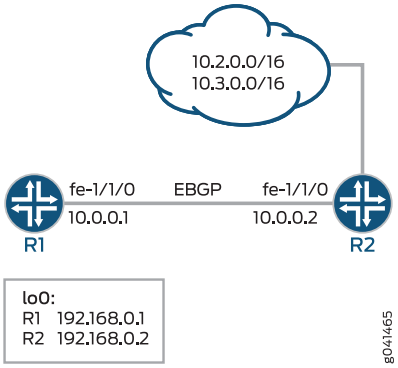
This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send two static routes to Device R1. On Device R1, an import policy specifies that the BGP-received routes can contain up to five communities to be considered a match. For example, 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 34 on page 352 shows the sample network.

Figure 34: BGP Policy with a Limit on the Number of Communities Accepted



Configuration

CLI Quick Configuration

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

Device 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
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the 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 *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

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.

Related Documentation

- *Example: Redistributing BGP Routes with a Specific Community Tag into IS-IS*
- *Understanding External BGP Peering Sessions*
- *BGP Configuration Overview*

Example: Defining a Routing Policy That Removes BGP Communities

This example shows how to create a policy that accepts BGP routes, but removes BGP communities from the routes.

- Requirements on page 358
- Overview on page 358
- Configuration on page 359
- Verification on page 364

Requirements

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

Overview

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:

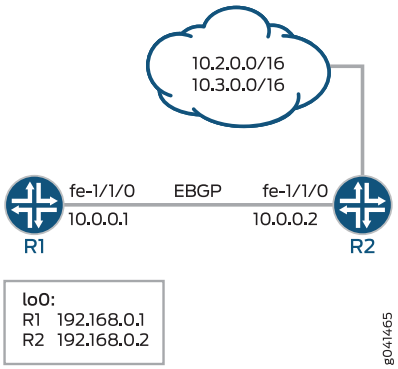
```
[edit policy-options]
policy-statement policy-name {
  term term-name {
    then community delete wild;
  }
}
```

To suppress a particular community from any autonomous system (AS), define the community as **community wild members "*:*community-value*"**.

Topology

Figure 35 on page 359 shows the sample network.

Figure 35: BGP Policy That Removes Communities



Configuration

CLI Quick Configuration

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

Device 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
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the 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 *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

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.

Related Documentation

- *Example: Redistributing BGP Routes with a Specific Community Tag into IS-IS*
- *Understanding External BGP Peering Sessions*
- *BGP Configuration Overview*

CHAPTER 22

Testing Policies

- Example: Testing a Routing Policy with Complex Regular Expressions on page 367

Example: Testing a Routing Policy with Complex Regular Expressions

This example shows how to test a routing policy using the **test policy** command to ensure that the policy produces the results that you expect before you apply it in a production environment. Regular expressions, especially complex ones, can be tricky to get right. This example shows how to use the **test policy** command to make sure that your regular expressions have the intended effect.

- Requirements on page 367
- Overview on page 367
- Configuration on page 369
- Verification on page 373

Requirements

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

Overview

This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send customer routes to Device R1. These static routes have multiple community values attached.

```
user@R2> show route match-prefix 172.16.* detail

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
```

```
                Communities: 64510:1 64510:10 64510:11 64510:100 64510:111

172.16.2.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:2 64510:20 64510:22 64510:200 64510:222

172.16.3.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:3 64510:30 64510:33 64510:300 64510:333

172.16.4.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:4 64510:40 64510:44 64510:400 64510:444
```

To test a complex regular expression, Device R2 has a policy called **test-regex** that locates routes. The policy is configured like this:

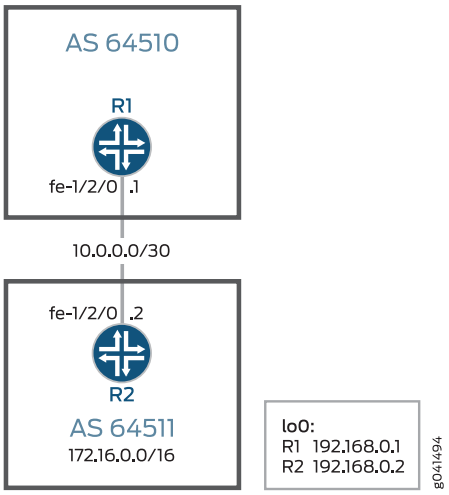
```
policy-statement test-regex {
  term find-routes {
    from community complex-regex;
    then accept;
  }
  term reject-the-rest {
    then reject;
  }
}
community complex-regex members "^64510:[13].*$";
```

This regular expression matches community values beginning with either 1 or 3.

Topology

Figure 36 on page 369 shows the sample network.

Figure 36: Routing Policy Test for Complex Regular Expressions



"CLI Quick Configuration" on page 369 shows the configuration for all of the devices in Figure 36 on page 369.

The section "Step-by-Step Procedure" on page 370 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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 peer-as 64511
set protocols bgp group ext neighbor 10.0.0.2
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 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 ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.0.0.1
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 then reject
set policy-options policy-statement test-regex term find-routes from community
  complex-regex
set policy-options policy-statement test-regex term find-routes then accept
set policy-options policy-statement test-regex term reject-the-rest then reject
```

```
set policy-options community complex-regex members "^64510:[13].*$"
set routing-options static route 172.16.1.0/24 reject
set routing-options static route 172.16.1.0/24 community 64510:1
set routing-options static route 172.16.1.0/24 community 64510:10
set routing-options static route 172.16.1.0/24 community 64510:11
set routing-options static route 172.16.1.0/24 community 64510:100
set routing-options static route 172.16.1.0/24 community 64510:111
set routing-options static route 172.16.2.0/24 reject
set routing-options static route 172.16.2.0/24 community 64510:2
set routing-options static route 172.16.2.0/24 community 64510:20
set routing-options static route 172.16.2.0/24 community 64510:22
set routing-options static route 172.16.2.0/24 community 64510:200
set routing-options static route 172.16.2.0/24 community 64510:222
set routing-options static route 172.16.3.0/24 reject
set routing-options static route 172.16.3.0/24 community 64510:3
set routing-options static route 172.16.3.0/24 community 64510:30
set routing-options static route 172.16.3.0/24 community 64510:33
set routing-options static route 172.16.3.0/24 community 64510:300
set routing-options static route 172.16.3.0/24 community 64510:333
set routing-options static route 172.16.4.0/24 reject
set routing-options static route 172.16.4.0/24 community 64510:4
set routing-options static route 172.16.4.0/24 community 64510:40
set routing-options static route 172.16.4.0/24 community 64510:44
set routing-options static route 172.16.4.0/24 community 64510:400
set routing-options static route 172.16.4.0/24 community 64510:444
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64511
```

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 R2:

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 lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set peer-as 64510
user@R2# set neighbor 10.0.0.1
```

3. Configure the routing policy that sends static routes.

```
[edit policy-options policy-statement send-static]
user@R2# set term 1 from protocol static
user@R2# set term 1 then accept
user@R2# set term 2 then reject
```

4. Configure the routing policy that tests a regular expression.

```
[edit policy-options policy-statement test-regex]
user@R2# set term find-routes from community complex-regex
user@R2# set term find-routes then accept
user@R2# set term reject-the-rest then reject

[edit policy-options community]
user@R2# set complex-regex members "^64510:[13].*$"
```

5. Configure the static routes and attaches community values.

```
[edit routing-options static route 172.16.1.0/24]
user@R2# set reject
user@R2# set community [ 64510:1 64510:10 64510:11 64510:100 64510:111 ]

[edit routing-options static route 172.16.2.0/24]
user@R2# set reject
user@R2# set community [ 64510:2 64510:20 64510:22 64510:200 64510:222 ]

[edit routing-options static route 172.16.3.0/24]
user@R2# set reject
user@R2# set community [ 64510:3 64510:30 64510:33 64510:300 64510:333 ]

[edit routing-options static route 172.16.4.0/24]
user@R2# set reject
user@R2# set community [ 64510:4 64510:40 64510:44 64510:400 64510:444 ]
```

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

   This affects Device R2's routing table, and as no impact on Device R1 and Device R3.

```
[edit routing-options ]
user@R2# set router-id 192.168.0.2
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.

```
user@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        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 ext {
        type external;
        peer-as 64510;
        neighbor 10.0.0.1;
    }
}

user@R2# show policy-options
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
    term 2 {
        then reject;
    }
}
policy-statement test-regex {
    term find-routes {
        from community complex-regex;
        then accept;
    }
    term reject-the-rest {
        then reject;
    }
}
community complex-regex members "^64510:[13].*$";

user@R2# show routing-options
static {
    route 172.16.1.0/24 {
        reject;
        community [ 64510:1 64510:10 64510:11 64510:100 64510:111 ];
    }
    route 172.16.2.0/24 {
        reject;
        community [ 64510:2 64510:20 64510:22 64510:200 64510:222 ];
    }
    route 172.16.3.0/24 {
        reject;
        community [ 64510:3 64510:30 64510:33 64510:300 64510:333 ];
    }
    route 172.16.4.0/24 {
        reject;
        community [ 64510:4 64510:40 64510:44 64510:400 64510:444 ];
    }
}
router-id 192.168.0.2;
autonomous-system 64511;
```

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

Verification

Confirm that the configuration is working properly.

Test to See Which Communities Match the Regular Expression

Purpose You can test the regular expression and its policy by using the test policy ***policy-name*** command.

Action

1. On Device R2, run the **test policy test-regex 0/0** command.

```
user@R2> test policy test-regex 0/0


inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[Static/5] 1d 00:32:50
                      Reject
172.16.3.0/24      *[Static/5] 1d 00:32:50
                      Reject

Policy test-regex: 2 prefix accepted, 5 prefix rejected
```

2. On Device R2, change the regular expression to match a community value containing any number of instances of the digit 2.

```
[edit policy-options community complex-regex]
user@R2# delete members "^64510:[13].*$"
user@R2# set members "^65020:2+$"
user@R2# commit
```

3. On Device R2, rerun the **test policy test-regex 0/0** command.

```
user@R2> test policy test-regex 0/0

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.2.0/24      *[Static/5] 1d 00:31:36
                      Reject

Policy test-regex: 1 prefix accepted, 6 prefix rejected
```

Meaning The 172.16.1.0 /24 and 172.16.3.0/24 routes both have communities attached that match the ^64510:[13].*$ expression. The 172.16.2.0/24 route has communities that match the ^65020:2+$ expression.

Related Documentation

- Understanding Routing Policy Tests on page 79
- Understanding How to Define BGP Communities and Extended Communities on page 68
- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 57

CHAPTER 23

Damp BGP Route Flapping

- Example: Configuring Damping Parameters on page 375
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 384

Example: Configuring Damping Parameters

This example shows how to configure damping parameters.

- Requirements on page 375
- Overview on page 375
- Configuration on page 376
- Verification on page 379

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.

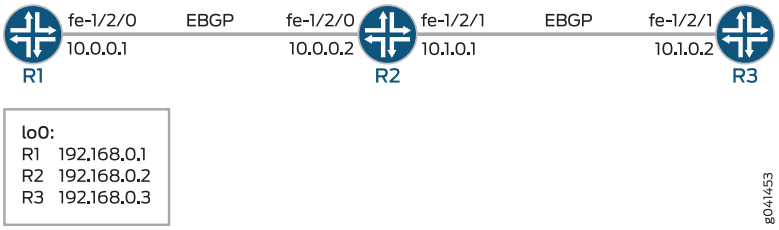
Device R2 damps routes received from Device R1 and Device R3 according to these criteria:

- 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 37 on page 376 shows the sample network.

Figure 37: BGP Flap Damping Topology



"CLI Quick Configuration" on page 376 shows the configuration for all of the devices in Figure 37 on page 376.

The section "Step-by-Step Procedure" on page 377 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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 224.0.0.0/7 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 *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

Confirm that the configuration is working properly.

- Causing Some Routes to Flap on page 380
- Checking the Route Flaps on page 380

- Verifying Route Flap Damping on page 381
- Displaying the Details of a Damped Route on page 381
- Verifying That Default Damping Parameters Are in Effect on page 382
- Filtering the Damping Information on page 383

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
224.0.0.0/7         [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
224.0.0.0/7 (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.

Related Documentation

- Understanding Damping Parameters on page 81
- Using Routing Policies to Damp BGP Route Flapping on page 82
- *Junos OS Feature Support Reference for SRX Series and J Series Devices*

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

This example shows how to configure an multiprotocol BGP multicast VPN (also called Next-Generation MVPN) with BGP route flap damping.

- Requirements on page 384
- Overview on page 384
- Configuration on page 385
- Verification on page 392

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.

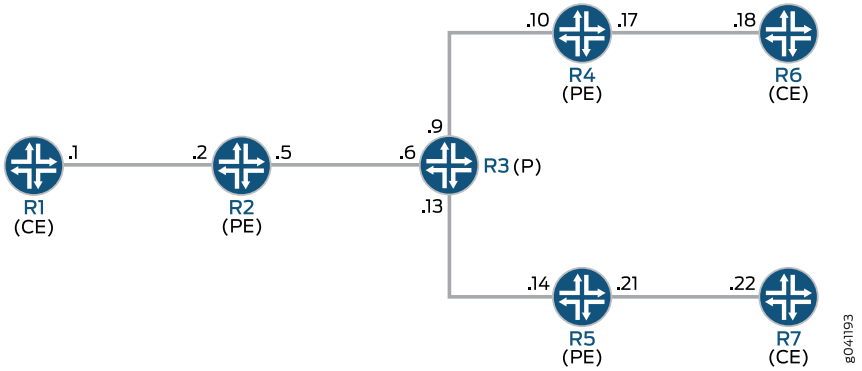
Overview

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.

Figure 38 on page 384 shows the topology used in this example.

Figure 38: 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 385 section. The "Configuring Device R4" on page 388 section shows the step-by-step configuration for PE Device R4.

Configuration

CLI Quick Configuration

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

Device 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 1.1.1.1/32
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 100.1.1.2
set protocols pim interface all
set routing-options router-id 1.1.1.1
```

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 1.1.1.2/32
set interfaces lo0 unit 102 family inet address 100.1.1.2/32
set protocols mpls interface ge-1/2/1.5
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 1.1.1.2
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 1.1.1.4
set protocols bgp group ibgp neighbor 1.1.1.5
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 100.1.1.2
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 1.1.1.2
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 1.1.1.3/32
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 1.1.1.3
```

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 1.1.1.4/32
set interfaces lo0 unit 104 family inet address 100.1.1.4/32
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 1.1.1.4
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 1.1.1.2 import dampPolicy
set protocols bgp group ibgp neighbor 1.1.1.5
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 100.1.1.2
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 1.1.1.4
set routing-options autonomous-system 1001
```

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 1.1.1.5/32
set interfaces lo0 unit 105 family inet address 100.1.1.5/32
set protocols mpls interface ge-1/2/0.14
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 1.1.1.5
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 1.1.1.2
set protocols bgp group ibgp neighbor 1.1.1.4
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 100.1.1.2
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 1.1.1.5
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 1.1.1.6/32
set protocols sap listen 224.1.1.1
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 100.1.1.2
```

```
set protocols pim interface all
set routing-options router-id 1.1.1.6
```

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 1.1.1.7/32
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 100.1.1.2
set protocols pim interface all
set routing-options router-id 1.1.1.7
```

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 *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 1.1.1.4/32
user@R4# set lo0 unit 104 family inet address 100.1.1.4/32
```

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 1.1.1.4
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 1.1.1.2 import dampPolicy
user@R4# set neighbor 1.1.1.5
```

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 100.1.1.2
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 1.1.1.4
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 1.1.1.4/32;
        }
    }
    unit 104 {
        family inet {
            address 100.1.1.4/32;
        }
    }
}
```

```
user@R4# show protocols
rsvp {
   interface all {
      aggregate;
   }
}
mpls {
   interface all;
   interface ge-1/2/0.10;
}
bgp {
   group ibgp {
      type internal;
      local-address 1.1.1.4;
      family inet-vpn {
         unicast;
         any;
      }
      family inet-mvpn {
         signaling {
            damping;
         }
      }
      neighbor 1.1.1.2 {
         import dampPolicy;
      }
      neighbor 1.1.1.5;
   }
}
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 100.1.1.2;
                }
            }
            interface ge-1/2/1.17 {
                mode sparse;
            }
        }
        mvpn;
    }
}
```

```
user@R4# show routing-optons
router-id 1.1.1.4;
autonomous-system 1001;
```

Verification

Confirm that the configuration is working properly.

- Verifying That Route Flap Damping Is Disabled on page 393
- Verifying Route Flap Damping on page 393

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...
1.1.1.2                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
1.1.1.5                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 1.1.1.2.

Related Documentation

- Understanding Damping Parameters on page 81
- Using Routing Policies to Damp BGP Route Flapping on page 82
- Example: Configuring Damping Parameters on page 375

CHAPTER 24

Source Class Usage and Destination Class Usage

- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 395

Example: Grouping Source and Destination Prefixes into a Forwarding Class

This example shows how to group source and destination prefixes into a forwarding class.

- Requirements on page 395
- Overview on page 395
- Configuration on page 398
- Verification on page 403

Requirements

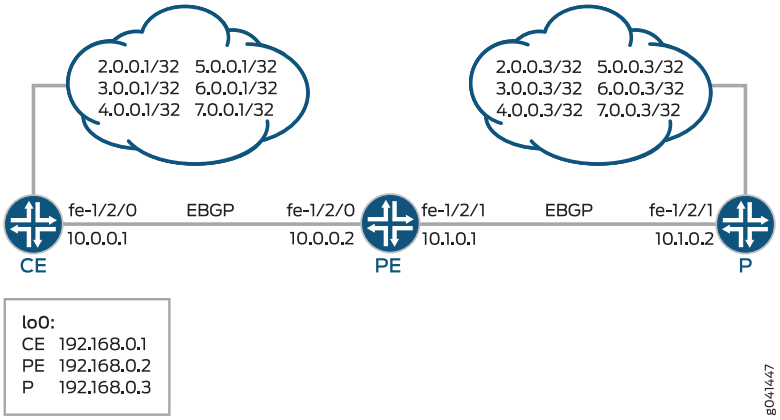
No special configuration beyond device initialization is required before configuring this example.

Overview

This example uses three routing devices: a customer edge (CE) device, a provider edge (PE) device, and a provider core (P) device.

Figure 39 on page 396 shows the sample network.

Figure 39: SCU and DCU Sample Network



Source class usage (SCU) counts packets sent to the customer edge by performing lookup on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge.

DCU counts packets from customers by performing a lookup of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

On Device PE's fe-1/2/1 interface, facing the provider core (represented by Device P), SCU input is configured with the **source-class-usage input** statement to track traffic originating at Device P and destined to Device CE. On this same interface, the **destination-class-usage input** statement is configured to track traffic originating at Device CE destined to the provider core.

```
user@PE# show interfaces fe-1/2/1 unit 0 family inet
accounting {
    source-class-usage {
        input; # tracks traffic destined to customer edge
    }
    destination-class-usage; # tracks traffic destined to provider core
}
address 10.1.0.1/30;
```

Unlike destination class usage (DCU), which only requires implementation on a single interface, accounting for SCU must be enabled on two interfaces: the inbound and outbound interfaces traversed by the source class. You must define explicitly the two interfaces on which SCU monitored traffic is expected to arrive and depart. This is because SCU performs two lookups in the routing table: a source address (SA) and a destination address (DA) lookup. In contrast, DCU only has a single destination address lookup.

On Device PE's fe-1/2/0 interface, facing Device CE, SCU output is configured with the **source-class-usage output** statement.

```
user@PE# show interfaces fe-1/2/0 unit 0 family inet
accounting {
    source-class-usage {
        output;
```

```
    }
  }
  address 10.0.0.2/30;
```

To account for traffic destined to the customer, the policy called scu_class uses route filters to place traffic into the gold1, gold2, and gold3 classes.

```
user@PE# show policy-options
policy-statement scu_class {
  term gold1 {
    from {
      route-filter 2.0.0.0/24 orlonger;
    }
    then source-class gold1;
  }
  term gold2 {
    from {
      route-filter 3.0.0.0/24 orlonger;
    }
    then source-class gold2;
  }
  term gold3 {
    from {
      route-filter 4.0.0.0/24 orlonger;
    }
    then source-class gold3;
  }
}
```

To account for traffic destined to the provider, the policy called dcu_class uses route filters to place traffic into the silver1, silver2, and silver3 classes.

```
user@PE# show policy-options
policy-statement dcu_class {
  term silver1 {
    from {
      route-filter 5.0.0.0/24 orlonger;
    }
    then destination-class silver1;
  }
  term silver2 {
    from {
      route-filter 6.0.0.0/24 orlonger;
    }
    then destination-class silver2;
  }
  term silver3 {
    from {
      route-filter 7.0.0.0/24 orlonger;
    }
    then destination-class silver3;
  }
}
```

The policies are then applied to the forwarding table.

```
  forwarding-table {
```

```
    export [ dcu_class scu_class ];
}
```

The example uses static routes to provide connectivity and loopback interface addresses to for testing the operation.

"CLI Quick Configuration" on page 398 shows the configuration for all of the devices in Figure 39 on page 396.

The section "Step-by-Step Procedure" on page 399 describes the steps on Device PE.

Configuration

CLI Quick Configuration

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

Device CE

```
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 interfaces lo0 unit 0 family inet address 2.0.0.1/32
set interfaces lo0 unit 0 family inet address 3.0.0.1/32
set interfaces lo0 unit 0 family inet address 4.0.0.1/32
set interfaces lo0 unit 0 family inet address 5.0.0.1/32
set interfaces lo0 unit 0 family inet address 6.0.0.1/32
set interfaces lo0 unit 0 family inet address 7.0.0.1/32
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 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 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 PE

```
set interfaces fe-1/2/0 unit 0 family inet accounting source-class-usage output
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 accounting source-class-usage input
set interfaces fe-1/2/1 unit 0 family inet accounting destination-class-usage
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 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 dcu_class term silver1 from route-filter 5.0.0.0/24
    orlonger
set policy-options policy-statement dcu_class term silver1 then destination-class silver1
set policy-options policy-statement dcu_class term silver2 from route-filter 6.0.0.0/24
    orlonger
set policy-options policy-statement dcu_class term silver2 then destination-class silver2
set policy-options policy-statement dcu_class term silver3 from route-filter 7.0.0.0/24
    orlonger
```

```
set policy-options policy-statement dcu_class term silver3 then destination-class silver3
set policy-options policy-statement scu_class term gold1 from route-filter 2.0.0.0/24
  orlonger
set policy-options policy-statement scu_class term gold1 then source-class gold1
set policy-options policy-statement scu_class term gold2 from route-filter 3.0.0.0/24
  orlonger
set policy-options policy-statement scu_class term gold2 then source-class gold2
set policy-options policy-statement scu_class term gold3 from route-filter 4.0.0.0/24
  orlonger
set policy-options policy-statement scu_class term gold3 then source-class gold3
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 200
set routing-options forwarding-table export dcu_class
set routing-options forwarding-table export scu_class
```

Device P

```
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 interfaces lo0 unit 0 family inet address 2.0.0.3/32
set interfaces lo0 unit 0 family inet address 3.0.0.3/32
set interfaces lo0 unit 0 family inet address 4.0.0.3/32
set interfaces lo0 unit 0 family inet address 5.0.0.3/32
set interfaces lo0 unit 0 family inet address 6.0.0.3/32
set interfaces lo0 unit 0 family inet address 7.0.0.3/32
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 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 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 static route 2.0.0.0/24 discard
set routing-options static route 3.0.0.0/24 discard
set routing-options static route 4.0.0.0/24 discard
set routing-options static route 5.0.0.0/24 discard
set routing-options static route 6.0.0.0/24 discard
set routing-options static route 7.0.0.0/24 discard
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To group source and destination prefixes in a forwarding class:

1. Create the router interfaces.

```
[edit interfaces]
user@PE# set fe-1/2/0 unit 0 family inet accounting source-class-usage output
user@PE# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30


user@PE# set fe-1/2/1 unit 0 family inet accounting source-class-usage input
```

```
user@PE# set fe-1/2/1 unit 0 family inet accounting destination-class-usage
user@PE# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

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

2. Configure BGP.

```
[edit protocols bgp group ext]
user@PE# set type external
user@PE# set export send-direct
user@PE# set neighbor 10.0.0.1 peer-as 100
user@PE# set neighbor 10.1.0.2 peer-as 300
```

3. Configure the DCU policy.

```
[edit policy-options policy-statement dcu_class]
user@PE# set term silver1 from route-filter 5.0.0.0/24 orlonger
user@PE# set term silver1 then destination-class silver1

user@PE# set term silver2 from route-filter 6.0.0.0/24 orlonger
user@PE# set term silver2 then destination-class silver2

user@PE# set term silver3 from route-filter 7.0.0.0/24 orlonger
user@PE# set term silver3 then destination-class silver3
```

4. Configure the SCU policy.

```
[edit policy-options policy-statement scu_class]
user@PE# set term gold1 from route-filter 2.0.0.0/24 orlonger
user@PE# set term gold1 then source-class gold1

user@PE# set term gold2 from route-filter 3.0.0.0/24 orlonger
user@PE# set term gold2 then source-class gold2

user@PE# set term gold3 from route-filter 4.0.0.0/24 orlonger
user@PE# set term gold3 then source-class gold3
```

5. Apply the policies to the forwarding table.

```
[edit routing-options forwarding-table]
user@PE# set export dcu_class
user@PE# set export scu_class
```

NOTE: You can refer to the same routing policy one or more times in the same or different export statement.

6. (Optional) Configure a routing policy that advertises direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@PE# set from protocol direct
user@PE# set then accept
```

7. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@PE# 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@PE# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          output;
        }
      }
      address 10.0.0.2/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          input;
        }
        destination-class-usage;
      }
      address 10.1.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.2/32;
    }
  }
}
```

```
user@PE# show protocols
bgp {
  group ext {
    type external;
    export send-direct;
    neighbor 10.0.0.1 {
      peer-as 100;
    }
    neighbor 10.1.0.2 {
      peer-as 300;
    }
  }
}
```

```
user@PE# show policy-options
policy-statement dcu_class {
  term silver1 {
    from {
      route-filter 5.0.0.0/24 orlonger;
    }
    then destination-class silver1;
  }
  term silver2 {
    from {
      route-filter 6.0.0.0/24 orlonger;
    }
    then destination-class silver2;
  }
  term silver3 {
    from {
      route-filter 7.0.0.0/24 orlonger;
    }
    then destination-class silver3;
  }
}
policy-statement scu_class {
  term gold1 {
    from {
      route-filter 2.0.0.0/24 orlonger;
    }
    then source-class gold1;
  }
  term gold2 {
    from {
      route-filter 3.0.0.0/24 orlonger;
    }
    then source-class gold2;
  }
  term gold3 {
    from {
      route-filter 4.0.0.0/24 orlonger;
    }
    then source-class gold3;
  }
}
policy-statement send-direct {
  term 1 {
    from protocol direct;
    then accept;
  }
}
```

```
user@PE# show routing-options
autonomous-system 200;
forwarding-table {
  export [ dcu_class scu_class ];
}
```

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

Verification

Confirm that the configuration is working properly.

- Making Sure That the DCU Policy Is Working on page 403
- Making Sure That the SCU Policy Is Working on page 403

Making Sure That the DCU Policy Is Working

Purpose Verify that traffic sent from the provider core into the customer network is causing the DCU policy counters to increment.

Action

1. From Device P, ping an address in the customer network.

```
user@P> ping rapid count 10000000 6.0.0.1

PING 6.0.0.1 (6.0.0.1): 56 data bytes
```

2. On Device PE, check the interface statistics on the interface facing the provider core.

```
user@PE> show interfaces statistics fe-1/2/1.0

  Logical interface fe-1/2/1.0 (Index 108) (SNMP ifIndex 546)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Input packets : 251956
    Output packets: 251961
    Protocol inet, MTU: 1500
      Flags: Sendbcast-pkt-to-re, DCU, SCU-in
                                                Packets                 Bytes
      Destination class            (packet-per-second)    (bits-per-second)

                           silver1                 7460                626640
                                   (                  0) (                 0)
                           silver2                22440               2401416
                                   (                256) (              171963)
                           silver3                 9004                756336
                                   (                  0) (                 0)
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.1.0.0/30, Local: 10.1.0.1, Broadcast: 10.1.0.3
```

Meaning Packet and bit rates are displayed with packet and byte counters.

Alternatively, you can use the **show interfaces destination-class all** command to display the same information.

Making Sure That the SCU Policy Is Working

Purpose Verify that traffic sent from the customer network into the provider core is causing the SCU policy counters to increment.

Action

1. From Device CE, ping an address in the customer network.

```
user@CE> ping rapid count 10000000 2.0.0.1
```

```
PING 6.0.0.1 (6.0.0.1): 56 data bytes
```

2. On Device PE, check the interface statistics on the interface facing the customer network.

```
user@PE> show interfaces statistics fe-1/2/0.0

  Logical interface fe-1/2/0.0 (Index 93) (SNMP ifIndex 554)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Input packets : 32246
    Output packets: 32245
    Protocol inet, MTU: 1500
      Flags: Sendbcast-pkt-to-re, Is-Primary, SCU-out
                                              Packets                 Bytes
      Source class                  (packet-per-second)    (bits-per-second)

                              gold1                  8871                745164
                                    (                 259) (             174497)
                              gold2                  1812                152208
                                    (                   0) (                  0)
                              gold3                  5711                479724
                                    (                   0) (                  0)
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.0.0.0/30, Local: 10.0.0.2, Broadcast: 10.0.0.3
```

Meaning Packet and bit rates are displayed with packet and byte counters.

Alternatively, you can use the **show interfaces source-class all** command to display the same information.

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 89
- Route Filter Match Conditions on page 115

CHAPTER 25

Conditional Routing Policies

- Example: Configuring Conditional Installation of Prefixes in a Routing Table on page 405

Example: Configuring Conditional Installation of Prefixes in a Routing Table

This example shows how to configure conditional installation of prefixes in a routing table using BGP export policy.

- Requirements on page 405
- Overview on page 405
- Configuration on page 408
- Verification on page 415

Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 9.0 or later

Overview

In this example, three routers in three different autonomous systems (ASs) are connected and configured with the BGP protocol. Router Internet, which is the upstream router, has five addresses configured on its lo0.0 loopback interface (11.1.1.1/32, 12.1.1.1/32, 13.1.1.1, 14.1.1.1/32, and 15.1.1.1/32), and an extra loopback address (192.168.9.1/32) to be 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 Router North.

Router North exports a default route into BGP, and advertises the default route and the five BGP routes to Router South, which is the downstream router. Router South receives the default route and only one other route (11.1.1.1/32), and installs this route and the default route in its routing table.

To summarize, the example meets the following requirements:

- On Device North, send 0/0 to Device South only if a particular route is also sent (in the example 11.1.1.1/32).
- On Device South, accept the default route and the 11.1.1.1/32 route. Drop all other routes. Consider that Device South might be receiving the entire Internet table, while the operator only wants Device South to have the default and one other specific prefix.

The first requirement is met with an export policy on Device North:

```
user@North# show policy-options
policy-statement conditional-export-bgp {
    term prefix_11 {
        from {
            protocol bgp;
            route-filter 11.0.0.0/5 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 {
        11.1.1.1/32;
        table inet.0;
    }
}
```

The logic of the conditional export policy can be summarized as follows: If 0/0 is present, and if 11.1.1.1/32 is present, then send the 0/0 prefix. This implies that if 11.1.1.1/32 is not present, then do not send 0/0.

The second requirement is met with an import policy on Device 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 11.0.0.0/8 orlonger;
        }
        then accept;
    }
    term 2 {
```



```
        then reject;
    }
}
```

In this example, four routes are dropped as a result of the import policy on Device South. This is because the export policy on Device North leaks all of the routes received from Device Internet, and the import policy on Device 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.

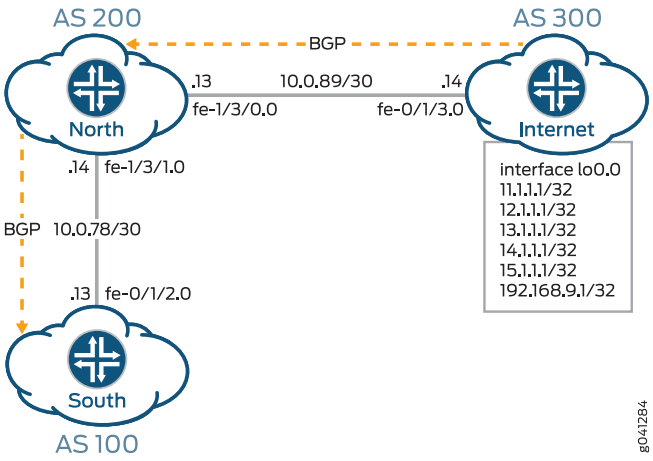
The rules of BGP route retention are as follows:

- 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.

Topology

Figure 40 on page 408 shows the topology used in this example.

Figure 40: Conditional Installation of Prefixes



Configuration

CLI Quick Configuration

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

Router Internet

```
set interfaces lo0 unit 0 family inet address 11.1.1.1/32
set interfaces lo0 unit 0 family inet address 12.1.1.1/32
set interfaces lo0 unit 0 family inet address 13.1.1.1/32
set interfaces lo0 unit 0 family inet address 14.1.1.1/32
set interfaces lo0 unit 0 family inet address 15.1.1.1/32
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 200
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 300
```

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 300
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 100
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
    11.0.0.0/5 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 11.1.1.1/32
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 200
```

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 200
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
    11.0.0.0/8 orlonger
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 100
```

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 11.1.1.1/32**
user@Internet# **set address 12.1.1.1/32**
user@Internet# **set address 13.1.1.1/32**
user@Internet# **set address 14.1.1.1/32**
user@Internet# **set address 15.1.1.1/32**
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 11.1.1.1/32**
   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), 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 11.0.0.0/5 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 11.0.0.0/8 orlonger**

```
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 200
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 300
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 100
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 200
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 300

Router North
[edit routing options]
user@North# set router-id 192.168.8.1
user@North# set autonomous-system 200

Router South
[edit routing options]
```

```
user@South# set router-id 192.168.7.1
user@South# set autonomous-system 100
```

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 11.1.1.1/32;
            address 12.1.1.1/32;
            address 13.1.1.1/32;
            address 14.1.1.1/32;
            address 15.1.1.1/32;
            address 192.168.9.1/32;
        }
    }
}

user@Internet# show protocols bgp
group toNorth {
    local-address 10.0.89.14;
    export into-bgp;
    peer-as 200;
    neighbor 10.0.89.13;
}

user@Internet# show policy-options
policy-statement into-bgp {
    term 1 {
        from interface lo0.3;
        then accept;
    }
}

user@Internet# show routing-options
router-id 192.168.9.1;
autonomous-system 300;
```

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 300;
  neighbor 10.0.89.14;
}
group toSouth {
  local-address 10.0.78.14;
  export conditional-export-bgp;
  peer-as 100;
  neighbor 10.0.78.13;
}

user@North# show policy-options
policy-statement conditional-export-bgp {
  term prefix_11 {
    from {
      protocol bgp;
      route-filter 11.0.0.0/5 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 {
    11.1.1.1/32;
    table inet.0;
  }
```

```
}

user@North# show routing-options
static {
  route 0.0.0.0/0 reject;
}
router-id 192.168.8.1;
autonomous-system 200;
```

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 200;
    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 11.0.0.0/8 orlonger;
      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 100;
```

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

Verification

Confirm that the configuration is working properly.

- Verifying BGP on page 415
- Verifying Prefix Advertisement from Router Internet to Router North on page 416
- Verifying Prefix Advertisement from Router North to Router South on page 417
- Verifying BGP Import Policy for Installation of Prefixes on page 418
- Verifying Conditional Export from Router North to Router South on page 418
- Verifying the Presence of Routes Hidden by Policy (Optional) on page 419

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 200  Local: 10.0.89.14+56187 AS 300
  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 200)
  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 300  Local: 10.0.89.13+179 AS 200
  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 300)
  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
* 11.1.1.1/32             Self                                    I
* 12.1.1.1/32             Self                                    I
* 13.1.1.1/32             Self                                    I
* 14.1.1.1/32             Self                                    I
* 15.1.1.1/32             Self                                    I
* 192.168.9.1/32          Self                                    I
```

   The output verifies that Router Internet advertises the routes 11.1.1.1/32, 12.1.1.1/32, 13.1.1.1/32, 14.1.1.1/32, 15.1.1.1/32, 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
* 11.1.1.1/32             10.0.89.14                              300 I
* 12.1.1.1/32             10.0.89.14                              300 I
* 13.1.1.1/32             10.0.89.14                              300 I
* 14.1.1.1/32             10.0.89.14                              300 I
* 15.1.1.1/32             10.0.89.14                              300 I
* 192.168.9.1/32          10.0.89.14                              300 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
* 11.1.1.1/32             Self                                    300 I
* 12.1.1.1/32             Self                                    300 I
* 13.1.1.1/32             Self                                    300 I
* 14.1.1.1/32             Self                                    300 I
* 15.1.1.1/32             Self                                    300 I
```

The output verifies that Router North is advertising the static route and the 11.1.1.1/32 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 11.1.1.1/32 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                              200 I
* 11.1.1.1/32             10.0.78.14                              200 300 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 11.1.1.1/32 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 11.1.1.1/32 route, Device North stops sending the default 0/0 route.

Action

1. Cause Device Internet to stop sending the 11.1.1.1/32 route by deactivating the 11.1.1.1/32 address on the loopback interface.

   ```
   [edit interfaces lo0 unit 0 family inet]
   user@Internet# deactivate address 11.1.1.1/32
   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
   * 12.1.1.1/32             Self                                    300 I
   * 13.1.1.1/32             Self                                    300 I
   ```

```
* 14.1.1.1/32              Self                                  300 I
* 15.1.1.1/32              Self                                  300 I
```

The output verifies that Router North is not advertising the default route to Router South. This is the expected behavior when the 11.1.1.1/32 route is not present.

3. Reactivate the 11.1.1.1/32 address on Device Internet's loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@Internet# activate address 11.1.1.1/32
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
  12.1.1.1/32             10.0.78.14                              200 300 I
  13.1.1.1/32             10.0.78.14                              200 300 I
  14.1.1.1/32             10.0.78.14                              200 300 I
  15.1.1.1/32             10.0.78.14                              200 300 I
```

The output verifies the presence of routes hidden by the import policy (12.1.1.1/32, 13.1.1.1/32, 14.1.1.1/32, and 15.1.1.1/32) 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                              200 I
```

```
* 11.1.1.1/32              10.0.78.14                      200 300 I
* 12.1.1.1/32              10.0.78.14                      200 300 I
* 13.1.1.1/32              10.0.78.14                      200 300 I
* 14.1.1.1/32              10.0.78.14                      200 300 I
* 15.1.1.1/32              10.0.78.14                      200 300 I
```

The output verifies the presence of previously hidden routes (12.1.1.1/32, 13.1.1.1/32, 14.1.1.1/32, and 15.1.1.1/32).

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.

Related Documentation

- Conditional Installation of Prefixes Use Cases on page 94
- Understanding Conditional Installation of Prefixes in a Routing Table on page 92

CHAPTER 26

Dynamic Routing Policies

- Example: Configuring Dynamic Routing Policies on page 421

Example: Configuring Dynamic Routing Policies

This example shows how to configure routing policy objects in a dynamic database that is not subject to the same verification required in the standard configuration database.

- Requirements on page 421
- Overview on page 421
- Configuration on page 422
- Verification on page 430

Requirements

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

Overview

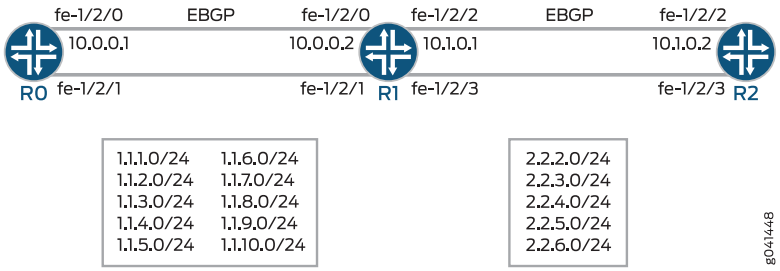
The verification process required to commit configuration changes can entail a significant amount of overhead and time.

The time it takes to commit changes to the dynamic database is much shorter than for the standard configuration database. You can reference these policies and policy objects in routing policies you configure in the standard database. BGP is the only protocol to which you can apply routing policies that reference policies and policy objects configured in the dynamic database. After you configure and commit a routing policy based on the objects configured in the dynamic database, you can quickly update any existing routing policy by making changes to the dynamic database configuration.

CAUTION: **Because Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.**

Figure 41 on page 422 shows the sample network.

Figure 41: Dynamic Routing Policy Sample Network



The example includes three routers with external BGP (EBGP) sessions established. Only Device R1 makes use of the dynamic database.

On Device R0's fe-1/2/1 interface, multiple IPv4 interfaces are configured, and a routing policy injects these prefixes into BGP, using the **from interface fe-1/2/1.0** policy condition as a shorthand method for specifying all of the IP addresses configured on Device R0's fe-1/2/1 interface.

Likewise, on Device R2's fe-1/2/3 interface, multiple IPv4 addresses are configured, and a routing policy injects these prefixes into BGP. Device R2's configuration is slightly different from Device R0's in that Device R2's configuration demonstrates the use of a prefix list.

On Device R1, in the dynamic database, two prefix lists are defined, one for the interface addresses learned from Device R0 and another for the interface addresses learned from Device R2. Device R1's standard database contains routing policies with prefix lists that are similar to those defined in the dynamic database.

In its peer session with Device R0, Device R1 has the static-database policies applied. In contrast, in its peer session with Device R2, Device R1's configuration references the dynamic database.

The results of these different configurations are analyzed in the "Verification" on page 430 section.

"CLI Quick Configuration" on page 422 shows the configuration for all of the devices in Figure 41 on page 422.

The section "Step-by-Step Procedure" on page 424 describes the steps on Device R1's dynamic database.

The section "Step-by-Step Procedure" on page 425 describes the steps on Device R1's standard database.

Configuration

CLI Quick Configuration

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

Device R0

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 1.1.4.1/24
```

```
set interfaces fe-1/2/1 unit 0 family inet address 1.1.3.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.2.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.1.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.5.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.6.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.7.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.8.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.9.1/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.10.1/24
set interfaces lo0 unit 0 family inet address 10.255.14.151/32
set protocols bgp group ext type external
set protocols bgp group ext neighbor 10.0.0.2 export t2
set protocols bgp group ext neighbor 10.0.0.2 peer-as 200
set policy-options policy-statement t2 from interface fe-1/2/0.0
set policy-options policy-statement t2 from interface fe-1/2/1.0
set policy-options policy-statement t2 then accept
set routing-options router-id 10.255.14.151
set routing-options autonomous-system 100
```

Device R1 Dynamic Database

```
[edit dynamic]
set policy-options prefix-list dyn_prfx1 1.1.1.0/24
set policy-options prefix-list dyn_prfx1 1.1.2.0/24
set policy-options prefix-list dyn_prfx1 1.1.3.0/24
set policy-options prefix-list dyn_prfx1 1.1.4.0/24
set policy-options prefix-list dyn_prfx1 1.1.5.0/24
set policy-options prefix-list dyn_prfx1 1.1.6.0/24
set policy-options prefix-list dyn_prfx1 1.1.7.0/24
set policy-options prefix-list dyn_prfx1 1.1.8.0/24
set policy-options prefix-list dyn_prfx2 2.2.2.0/24
set policy-options prefix-list dyn_prfx2 2.2.3.0/24
set policy-options prefix-list dyn_prfx2 2.2.4.0/24
set policy-options prefix-list dyn_prfx2 2.2.5.0/24
set policy-options prefix-list dyn_prfx2 2.2.6.0/24
set policy-options policy-statement dyn_policy1 term t1 from prefix-list dyn_prfx1
set policy-options policy-statement dyn_policy1 term t1 then accept
set policy-options policy-statement dyn_policy1 term t2 then reject
set policy-options policy-statement dyn_policy2 term t1 from prefix-list dyn_prfx2
set policy-options policy-statement dyn_policy2 term t1 then accept
set policy-options policy-statement dyn_policy2 term t2 then reject
```

Device R1 Standard Database

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 1.1.4.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.3.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.2.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.1.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.5.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.6.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.7.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.8.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.9.2/24
set interfaces fe-1/2/1 unit 0 family inet address 1.1.10.2/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.2.2/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.3.2/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.4.2/24
```

```
set interfaces fe-1/2/3 unit 0 family inet address 2.2.5.2/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.6.2/24
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group to_r0 idle-after-switch-over 300
set protocols bgp group to_r0 neighbor 10.0.0.1 import dyn_policy1
set protocols bgp group to_r0 neighbor 10.0.0.1 export dyn_policy2
set protocols bgp group to_r0 neighbor 10.0.0.1 peer-as 100
set protocols bgp group to_R2 import static_policy1
set protocols bgp group to_R2 export static_policy2
set protocols bgp group to_R2 idle-after-switch-over 300
set protocols bgp group to_R2 neighbor 10.1.0.2 peer-as 300
set policy-options prefix-list static_prfx1 2.2.2.0/24
set policy-options prefix-list static_prfx1 2.2.3.0/24
set policy-options prefix-list static_prfx1 2.2.4.0/24
set policy-options prefix-list static_prfx1 2.2.5.0/24
set policy-options prefix-list static_prfx2 1.1.1.0/24
set policy-options prefix-list static_prfx2 1.1.2.0/24
set policy-options prefix-list static_prfx2 1.1.3.0/24
set policy-options prefix-list static_prfx2 1.1.4.0/24
set policy-options policy-statement dyn_policy1 dynamic-db
set policy-options policy-statement dyn_policy2 dynamic-db
set policy-options policy-statement static_policy1 term t1 from prefix-list static_prfx1
set policy-options policy-statement static_policy1 term t1 then accept
set policy-options policy-statement static_policy1 term t2 then reject
set policy-options policy-statement static_policy2 term t1 from prefix-list static_prfx2
set policy-options policy-statement static_policy2 term t1 then accept
set policy-options policy-statement static_policy2 term t2 then reject
set routing-options autonomous-system 200
```

Device R2

```
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/3 unit 0 family inet address 2.2.2.1/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.3.1/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.4.1/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.5.1/24
set interfaces fe-1/2/3 unit 0 family inet address 2.2.6.1/24
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group to_vin neighbor 10.1.0.1 export p1
set protocols bgp group to_vin neighbor 10.1.0.1 peer-as 200
set policy-options prefix-list ppx1 2.2.2.0/24
set policy-options prefix-list ppx1 2.2.3.0/24
set policy-options prefix-list ppx1 2.2.4.0/24
set policy-options prefix-list ppx1 2.2.5.0/24
set policy-options prefix-list ppx1 2.2.6.0/24
set policy-options policy-statement p1 term t1 from family inet
set policy-options policy-statement p1 term t1 from prefix-list ppx1
set policy-options policy-statement p1 term t1 then accept
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1's dynamic database:

1. Enter configuration mode for the dynamic database.

```
user@R1> configure dynamic
Entering configuration mode
[edit dynamic]
```

2. Create a prefix list for the interface addresses learned from Device R0.

```
[edit dynamic policy-options prefix-list dyn_prfx1]
user@R1# set 1.1.1.0/24
user@R1# set 1.1.2.0/24
user@R1# set 1.1.3.0/24
user@R1# set 1.1.4.0/24
user@R1# set 1.1.5.0/24
user@R1# set 1.1.6.0/24
user@R1# set 1.1.7.0/24
user@R1# set 1.1.8.0/24
```

3. Create a prefix list for the interface addresses learned from Device R2.

```
[edit dynamic policy-options prefix-list dyn_prfx2]
user@R1# set 2.2.2.0/24
user@R1# set 2.2.3.0/24
user@R1# set 2.2.4.0/24
user@R1# set 2.2.5.0/24
user@R1# set 2.2.6.0/24
```

4. Configure the routing policies.

```
[edit dynamic policy-options policy-statement dyn_policy1]
user@R1# set term t1 from prefix-list dyn_prfx1
user@R1# set term t1 then accept
user@R1# set term t2 then reject

user@R1# set term t1 from prefix-list dyn_prfx2
user@R1# set term t1 then accept
user@R1# set term t2 then reject
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1's standard database:

1. Create the router interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R1# set fe-1/2/2 unit 0 family inet address 10.1.0.1/30

user@R1# set fe-1/2/1 unit 0 family inet address 1.1.4.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.3.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.2.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.1.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.5.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.6.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.7.2/24
```

```
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.8.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.9.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 1.1.10.2/24

user@R1# set fe-1/2/3 unit 0 family inet address 2.2.2.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 2.2.3.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 2.2.4.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 2.2.5.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 2.2.6.2/24

user@R1# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Create routing policies that reference the policies in the dynamic database.

```
[edit policy-options]
user@R1# set policy-statement dyn_policy1 dynamic-db
user@R1# set policy-statement dyn_policy2 dynamic-db
```

3. Configure BGP peering with Device R0.

```
[edit protocols bgp group to_r0]
user@R1# set neighbor 10.0.0.1 peer-as 100
```

4. Apply the dynamic database policies to the BGP peering with Device R0.

```
[edit protocols bgp group to_r0]
user@R1# set neighbor 10.0.0.1 import dyn_policy1
user@R1# set neighbor 10.0.0.1 export dyn_policy2
```

5. Configure a prefix list for prefixes learned from Device R0.

```
[edit policy-options prefix-list static_prfx2]
user@R1# set 1.1.1.0/24
user@R1# set 1.1.2.0/24
user@R1# set 1.1.3.0/24
user@R1# set 1.1.4.0/24
```

6. Configure a prefix list for prefixes learned from Device R2.

```
[edit policy-options prefix-list static_prfx1]
user@R1# set 2.2.2.0/24
user@R1# set 2.2.3.0/24
user@R1# set 2.2.4.0/24
user@R1# set 2.2.5.0/24
```

7. Configure the static database policies.

```
[edit policy-options policy-statement static_policy1]
user@R1# set term t1 from prefix-list static_prfx1
user@R1# set term t1 then accept
user@R1# set term t2 then reject

[edit policy-options policy-statement static_policy2]
user@R1# set term t1 from prefix-list static_prfx2
user@R1# set term t1 then accept
user@R1# set term t2 then reject
```

8. Configure BGP peering with Device R2.

```
[edit protocols bgp group to_R2]
user@R1# set neighbor 10.1.0.2 peer-as 300
```

9. Apply the static database policies to the BGP peering with Device R2.

```
[edit protocols bgp group to_R2]
user@R1# set import static_policy1
user@R1# set export static_policy2
```

10. (Optional) Configure the router not to reestablish the BGP peering sessions after an active nonstop routing switchover either for a specified period or until you manually reestablish the session.

    This statement is particularly useful with dynamic routing policies because the dynamic database is not synchronized with the backup Routing Engine when nonstop active routing (NSR) is enabled. As a result, if a switchover to a backup Routing Engine occurs, import and export policies running on the master Routing Engine at the time of the switchover might no longer be available. Therefore, you might want to prevent a BGP peering session from automatically being reestablished as soon as a switchover occurs.

```
[edit protocols bgp]
user@R1# set group to_r0 idle-after-switch-over 300
user@R1# set group to_R2 idle-after-switch-over 300
```

11. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set routing-options autonomous-system 200
```

Results Confirm your configuration by entering the **show** command from configuration mode in the dynamic database, and the **show interfaces**, **show protocols**, **show policy-options** and **show routing-options** commands from configuration mode in the standard database. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1 Dynamic

```
[edit dynamic]
user@R1# show
policy-options {
  prefix-list dyn_prfx1 {
    1.1.1.0/24;
    1.1.2.0/24;
    1.1.3.0/24;
    1.1.4.0/24;
    1.1.5.0/24;
    1.1.6.0/24;
    1.1.7.0/24;
    1.1.8.0/24;
  }
  prefix-list dyn_prfx2 {
    2.2.2.0/24;
    2.2.3.0/24;
    2.2.4.0/24;
    2.2.5.0/24;
```

```
    2.2.6.0/24;
  }
  policy-statement dyn_policy1 {
    term t1 {
      from {
        prefix-list dyn_prfx1;
      }
      then accept;
    }
    term t2 {
      then reject;
    }
  }
  policy-statement dyn_policy2 {
    term t1 {
      from {
        prefix-list dyn_prfx2;
      }
      then accept;
    }
    term t2 {
      then reject;
    }
  }
}
```

Device R1 Standard

```
[edit]
user@R1# 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 1.1.4.2/24;
      address 1.1.3.2/24;
      address 1.1.2.2/24;
      address 1.1.1.2/24;
      address 1.1.5.2/24;
      address 1.1.6.2/24;
      address 1.1.7.2/24;
      address 1.1.8.2/24;
      address 1.1.9.2/24;
      address 1.1.10.2/24;
    }
  }
}
fe-1/2/2 {
  unit 0 {
    family inet {
      address 10.1.0.1/30;
    }
```

```
    }
  }
  fe-1/2/3 {
    unit 0 {
      family inet {
        address 2.2.2.2/24;
        address 2.2.3.2/24;
        address 2.2.4.2/24;
        address 2.2.5.2/24;
        address 2.2.6.2/24;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 192.168.0.2/32;
      }
    }
  }

  user@R1# show protocols
  bgp {
    group to_r0 {
      idle-after-switch-over 300;
      neighbor 10.0.0.1 {
        import dyn_policy1;
        export dyn_policy2;
        peer-as 100;
      }
    }
    group to_R2 {
      import static_policy1;
      export static_policy2;
      idle-after-switch-over 300;
      neighbor 10.1.0.2 {
        peer-as 300;
      }
    }
  }
  user@R1# show policy-options
  prefix-list static_prfx1 {
    2.2.2.0/24;
    2.2.3.0/24;
    2.2.4.0/24;
    2.2.5.0/24;
  }
  prefix-list static_prfx2 {
    1.1.1.0/24;
    1.1.2.0/24;
    1.1.3.0/24;
    1.1.4.0/24;
  }
  policy-statement dyn_policy1 {
    dynamic-db;
```

```
}
policy-statement dyn_policy2 {
  dynamic-db;
}
policy-statement static_policy1 {
  term t1 {
    from {
      prefix-list static_prfx1;
    }
    then accept;
  }
  term t2 {
    then reject;
  }
}
policy-statement static_policy2 {
  term t1 {
    from {
      prefix-list static_prfx2;
    }
    then accept;
  }
  term t2 {
    then reject;
  }
}

user@R1# show routing-options
autonomous-system 200;
```

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

Verification

Confirm that the configuration is working properly.

- Checking the Configured Policies on Device R1 on page 430
- Checking the Routes Advertised from Device R0 to Device R1 on page 431
- Checking the Routes That Device R1 Is Receiving from Device R0 on page 432
- Checking the Routes Advertised from Device R2 to Device R1 on page 432
- Checking the Routes That Device R1 Is Receiving from Device R2 on page 433
- Checking the Routes That Device R1 Is Advertising to Device R0 on page 433
- Checking the Routes That Device R1 Is Advertising to Device R2 on page 434

Checking the Configured Policies on Device R1

Purpose Verify that Device R1 has the dynamic and static policies in effect.

Action From Device R1, enter the **show policy** command.

```
user@R1> show policy
Configured policies:
dyn_policy1
```

```
dyn_policy2
static_policy1
static_policy2
dyn_policy1
dyn_policy2
```

Meaning The dynamic policies are listed two times because they are configured two times, the first and central configuration in the dynamic database. The secondary configuration is in the static database, where the dynamic database is referenced, as shown here:

Configured in the Dynamic Database

```
policy-statement dyn_policy1 {
  term t1 {
    from {
      prefix-list dyn_prfx1;
    }
    then accept;
  }
  term t2 {
    then reject;
  }
}
policy-statement dyn_policy2 {
  term t1 {
    from {
      prefix-list dyn_prfx2;
    }
    then accept;
  }
  term t2 {
    then reject;
  }
}
```

Referenced from the Static Database

```
policy-statement dyn_policy1 {
  dynamic-db;
}
policy-statement dyn_policy2 {
  dynamic-db;
}
```

Checking the Routes Advertised from Device R0 to Device R1

Purpose Verify that Device R0's routing policy is working.

Action From Device R0, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R1.

```
user@R0> show route advertising-protocol bgp 10.0.0.2
inet.0: 28 destinations, 28 routes (28 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 1.1.1.0/24              Self                                    I
* 1.1.2.0/24              Self                                    I
* 1.1.3.0/24              Self                                    I
* 1.1.4.0/24              Self                                    I
* 1.1.5.0/24              Self                                    I
* 1.1.6.0/24              Self                                    I
```

```
* 1.1.7.0/24             Self                                I
* 1.1.8.0/24             Self                                I
* 1.1.9.0/24             Self                                I
* 1.1.10.0/24            Self                                I
* 10.0.0.0/30            Self                                I
```

Meaning Device R0 is sending the expected routes to Device R1.

Checking the Routes That Device R1 Is Receiving from Device R0

Purpose Verify that Device R1's import routing policy is working.

Action From Device R1, enter the **show route receive-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route receive-protocol bgp 10.0.0.1
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  1.1.1.0/24              10.0.0.1                                100 I
  1.1.2.0/24              10.0.0.1                                100 I
  1.1.3.0/24              10.0.0.1                                100 I
  1.1.4.0/24              10.0.0.1                                100 I
  1.1.5.0/24              10.0.0.1                                100 I
  1.1.6.0/24              10.0.0.1                                100 I
  1.1.7.0/24              10.0.0.1                                100 I
  1.1.8.0/24              10.0.0.1                                100 I
```

Meaning Some of the routes that are sent by Device R0 are not received by Device R1. The routes 1.1.9.0/24, 1.1.10.0/24, and 10.0.0.0/30 are missing. This is because Device R1's import policy, applied to the BGP peering session with Device R0 using the **import dyn_policy1** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list dyn_prfx1 {
  1.1.1.0/24;
  1.1.2.0/24;
  1.1.3.0/24;
  1.1.4.0/24;
  1.1.5.0/24;
  1.1.6.0/24;
  1.1.7.0/24;
  1.1.8.0/24;
}
```

Checking the Routes Advertised from Device R2 to Device R1

Purpose Verify that Device R2's routing policy is working.

Action From Device R2, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R1.

```
user@R2> show route advertising-protocol bgp 10.1.0.1
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 2.2.2.0/24              Self                                    I
* 2.2.3.0/24              Self                                    I
* 2.2.4.0/24              Self                                    I
```

```
* 2.2.5.0/24              Self                                  I
* 2.2.6.0/24              Self                                  I
```

Meaning Device R2 is sending the expected routes to Device R1.

Checking the Routes That Device R1 Is Receiving from Device R2

Purpose Verify that Device R1's import routing policy is working.

Action From Device R1, enter the **show route receive-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route receive-protocol bgp 10.1.0.2
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  2.2.2.0/24              10.1.0.2                                300 I
  2.2.3.0/24              10.1.0.2                                300 I
  2.2.4.0/24              10.1.0.2                                300 I
  2.2.5.0/24              10.1.0.2                                300 I
```

Meaning One of the routes that is sent by Device R2 is not received by Device R1. The route 2.2.6.0/24 is missing. This is because Device R1's import policy, applied to the BGP peering session with Device R2 using the **import static_policy1** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list static_prfx1 {
  2.2.2.0/24;
  2.2.3.0/24;
  2.2.4.0/24;
  2.2.5.0/24;
}
```

Checking the Routes That Device R1 Is Advertising to Device R0

Purpose Verify that Device R1's export routing policy is working.

Action From Device R1, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route advertising-protocol bgp 10.0.0.1
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 2.2.2.0/24              Self                                    I
* 2.2.3.0/24              Self                                    I
* 2.2.4.0/24              Self                                    I
* 2.2.5.0/24              Self                                    I
* 2.2.6.0/24              Self                                    I
```

Meaning Perhaps unexpectedly, the route that Device R1 did not receive through BGP from Device R2 (2.2.6.0/24) is nonetheless being advertised by Device R1 through BGP to Device R0. This is happening for two reasons. The first reason is that route 2.2.6.0/24 is in Device R1's routing table, albeit as a direct route, as shown here:

```
user@R1> show route 2.2.6.0/24 protocol direct
```

```
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
+ = Active Route, - = Last Active, * = Both

2.2.6.0/24         *[Direct/0] 2d 22:51:41
                    > via fe-1/2/3.0
```

The second reason is that Device R1's export policy, applied to the BGP peering session with Device R0 using the **export dyn_policy2** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list dyn_prfx2 {
  2.2.2.0/24;
  2.2.3.0/24;
  2.2.4.0/24;
  2.2.5.0/24;
  2.2.6.0/24;
}
```

Note the inclusion of 2.2.6.0/24.

Checking the Routes That Device R1 Is Advertising to Device R2

Purpose Verify that Device R1's export routing policy is working.

Action From Device R1, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R2.

```
user@R1> show route advertising-protocol bgp 10.1.0.2
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 1.1.1.0/24              Self                                    I
* 1.1.2.0/24              Self                                    I
* 1.1.3.0/24              Self                                    I
* 1.1.4.0/24              Self                                    I
```

Meaning Device R1 is sending the expected routes to Device R2. Device R1's export policy, applied to the BGP peering session with Device R2 using the **export static_policy2** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list static_prfx2 {
  1.1.1.0/24;
  1.1.2.0/24;
  1.1.3.0/24;
  1.1.4.0/24;
}
```

Related Documentation

- Understanding Dynamic Routing Policies on page 97
- Example: Configuring Routing Policy Prefix Lists on page 261

CHAPTER 27

Discard Routing Policy

- Example: Forwarding Packets to the Discard Interface on page 435

Example: Forwarding Packets to the Discard Interface

This example shows how to use discard routing to mitigate denial of service (DoS) attacks, protect vital network resources from outside attack, provide protection services for customers so that each customer can initiate its own protection, and log and track DoS attempts..

- Requirements on page 435
- Overview on page 435
- Configuration on page 438
- Verification on page 442

Requirements

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

Overview

In discard routing, routers are configured with rules that disallow millions of requests in a short period of time from being sent to the same address. If too many requests are received in a short period of time, the router simply discards the requests without forwarding them. The requests are sent to a router that does not forward the packets. The problematic routes are sometimes referred to as discard routes or black-holed routes. The types of routes that should be discarded are identified as attacks to customers from peers or other customers, attacks from customers to peers or other customers, attack controllers, which are hosts providing attack instructions, and unallocated address spaces, known as bogons or invalid IP addresses.

After the attack attempt is identified, operators can put a configuration in place to mitigate the attack. One way to configure discard routing in Junos OS is to create a discard static route for each next hop used for discard routes. A discard static route uses the **discard** option.

For example:

```
user@host# show routing-options
static {
  route 192.0.2.101/32 discard;
  route 192.0.2.103/32 discard;
  route 192.0.2.105/32 discard;
}
```

```
user@host> show route protocol static terse
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 192.0.2.101/32     S   5                        Discard
* ? 192.0.2.103/32     S   5                        Discard
* ? 192.0.2.105/32     S   5                        Discard
```

Another strategy, which is the main focus of this example, is to use routing policy and the discard interface. In this approach, the discard interface contains the next hop you are assigning to the black-hole routes. A discard interface can have only one logical unit (unit 0), but you can configure multiple IP addresses on unit 0.

For example:

```
user@host# show interfaces dsc
unit 0 {
  family inet {
    address 192.0.2.102/32 {
      destination 192.0.2.101;
    }
    address 192.0.2.104/32 {
      destination 192.0.2.103;
    }
    address 192.0.2.106/32 {
      destination 192.0.2.105;
    }
  }
}
```

```
user@host> show interfaces terse dsc
b
Interface               Admin Link Proto    Local                 Remote
dsc                     up    up
dsc.0                   up    up   inet     192.0.2.102         --> 192.0.2.101
                                            192.0.2.104         --> 192.0.2.103
                                            192.0.2.106         --> 192.0.2.105
```

The advantage of using a discard interface instead of using discard static routes is that the discard interface allows you to configure and assign filters to the interface for counting, logging, and sampling the traffic. This is demonstrated in this example.

To actually discard packets requires a routing policy attached to the BGP sessions. To locate discard-eligible routes, you can use a route filter, an access list, or a BGP community value.

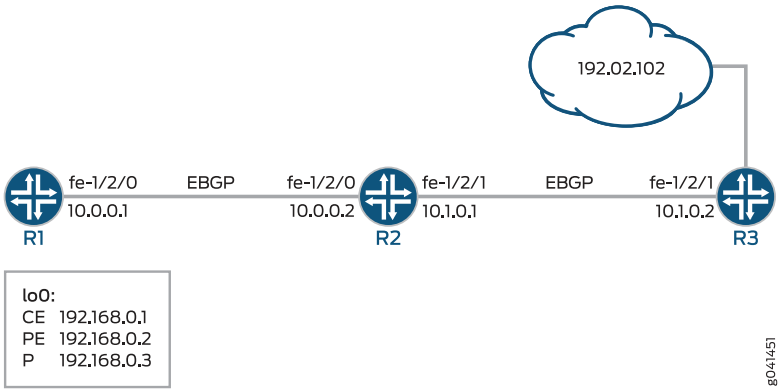
For example, here is how you would use a route filter:

Route Filter

```
protocols {
  bgp {
    import blackhole-by-route;
  }
}
policy-options {
  policy-statement blackhole-by-route {
    term specific-routes {
      from {
        route-filter 10.10.10.1/32 exact;
        route-filter 10.20.20.2/32 exact;
        route-filter 10.30.30.3/32 exact;
        route-filter 10.40.40.4/32 exact;
      }
      then {
        next-hop 192.0.2.101
      }
    }
  }
}
```

Figure 42 on page 437 shows the sample network.

Figure 42: Discard Interface Sample Network



The example includes three routers with external BGP (EBGP) sessions established.

Device R1 represents the attacking device. Device R3 represents the router closest to the device that is being attacked. Device R2 mitigates the attack by forwarding packets to the discard interface.

The example shows an outbound filter applied to the discard interface.

NOTE: An issue with using a single black-hole filter is visibility. All discard packets increment the same counter. To see which categories of packets are being discarded, use destination class usage (DCU), and associate a user-defined class with each black-hole community. Then reference the DCU classes in a firewall filter. For related examples, see "Example: Grouping Source and Destination Prefixes into a Forwarding Class" on page 395 and *Example: Configuring a Rate-Limiting Filter Based on Destination Class*.

Compared to using route filters and access lists, using a community value is the least administratively difficult and the most scalable approach. Therefore, this is the approach shown in this example.

By default, the next hop must be equal the external BGP (EBGP) peer address. Altering the next hop for black-hole services requires the multihop feature to be configured on the EBGP sessions.

"CLI Quick Configuration" on page 438 shows the configuration for all of the devices in Figure 42 on page 437.

The section "Step-by-Step Procedure" on page 439 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device 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 peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
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 dsc unit 0 family inet filter output log-discard
set interfaces dsc unit 0 family inet address 192.0.2.102/32 destination 192.0.2.101
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp import blackhole-policy
set protocols bgp group ext type external
set protocols bgp group ext multihop
set protocols bgp group ext export dsc-export
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 blackhole-policy term blackhole-communities from
    community blackhole-all-routers
set policy-options policy-statement blackhole-policy term blackhole-communities then
    next-hop 192.0.2.101
set policy-options policy-statement dsc-export from route-filter 192.0.2.101/32 exact
set policy-options policy-statement dsc-export from route-filter 192.0.2.102/32 exact
set policy-options policy-statement dsc-export then community set blackhole-all-routers
set policy-options policy-statement dsc-export then accept
set policy-options community blackhole-all-routers members 100:5555
set routing-options static route 192.0.2.102/32 next-hop 192.0.2.101
set routing-options autonomous-system 200
set firewall filter log-discard term one then count counter
set firewall filter log-discard term one then log
```

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 interfaces lo0 unit 0 family inet address 192.0.2.102/32
```

```
set protocols bgp group ext type external
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.1.0.1
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Create the router 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 a firewall filter that matches all packets and counts and logs the packets.

```
[edit firewall filter log-discard term one]
user@R2# set then count counter
user@R2# set then log
```

3. Create a discard interface and apply the output firewall filter.

   Input firewall filters have no impact in this context.

```
[edit interfaces dsc unit 0 family inet]
user@R2# set filter output log-discard
user@R2# set address 192.0.2.102/32 destination 192.0.2.101
```

4. Configure a static route that sends the next hop to the destination address that is specified in the discard interface.

```
[edit routing-options static]
user@R2# set route 192.0.2.102/32 next-hop 192.0.2.101
```

5. Configure BGP peering.

```
[edit protocols bgp ]
user@R2# set group ext type external
user@R2# set group ext multihop
user@R2# set group ext neighbor 10.0.0.1 peer-as 100
user@R2# set group ext neighbor 10.1.0.2 peer-as 300
```

6. Configure the routing policies.

```
[edit policy-options policy-statement blackhole-policy term blackhole-communities]
user@R2# set from community blackhole-all-routers
user@R2# set then next-hop 192.0.2.101

[edit policy-options policy-statement dsc-export]
user@R2# set from route-filter 192.0.2.101/32 exact
user@R2# set from route-filter 192.0.2.102/32 exact
```

```
user@R2# set then community set blackhole-all-routers
user@R2# set then accept


[edit policy-options community blackhole-all-routers]
user@R2# set members 100:5555
```

7. Apply the routing policies.

```
[edit protocols bgp ]
user@R2# set import blackhole-policy
user@R2# set group ext export dsc-export
```

8. 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**, **show routing-options**, and **show firewall** 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
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;
        }
    }
}
dsc {
    unit 0 {
        family inet {
            filter {
                output log-discard;
            }
            address 192.0.2.102/32 {
                destination 192.0.2.101;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
```

```
}
```

```
user@R2# show protocols
bgp {
  import blackhole-policy;
  group ext {
    type external;
    multihop;
    export dsc-export;
    neighbor 10.0.0.1 {
      peer-as 100;
    }
    neighbor 10.1.0.2 {
      peer-as 300;
    }
  }
}
```

```
user@R2# show policy-options
policy-statement blackhole-policy {
  term blackhole-communities {
    from community blackhole-all-routers;
    then {
      next-hop 192.0.2.101;
    }
  }
}
policy-statement dsc-export {
  from {
    route-filter 192.0.2.101/32 exact;
    route-filter 192.0.2.102/32 exact;
  }
  then {
    community set blackhole-all-routers;
    accept;
  }
}
community blackhole-all-routers members 100:5555;
```

```
user@R2# show routing-options
static {
  route 192.0.2.102/32 next-hop 192.0.2.101;
}
autonomous-system 200;
```

```
user@R2# show firewall
filter log-discard {
  term one {
    then {
      count counter;
      log;
    }
  }
}
```

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

Verification

Confirm that the configuration is working properly.

- Clearing the Firewall Counters on page 442
- Pinging the 192.0.2.101 Address on page 442
- Checking the Output Filter on page 442
- Checking the Community Attribute on page 443

Clearing the Firewall Counters

Purpose Clear the counters to make sure you are starting from a known zero (0) state.

Action

1. From Device R2, run the **clear firewall** command.

   ```
   user@R2> clear firewall filter log-discard
   ```

2. From Device R2, run the **show firewall** command.

   ```
   user@R2> show firewall filter log-discard
   Filter: /log-discard
   Counters:
   Name                                                Bytes              Packets
   counter                                                 0
   0
   ```

Pinging the 192.0.2.101 Address

Purpose Send packets to the destination address.

Action From Device R1, run the **ping** command.

```
user@R1> ping 192.0.2.101
PING 192.0.2.101 (192.0.2.101): 56 data bytes
^C
--- 192.0.2.101 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
```

Meaning As expected, the ping request fails, and no response is sent. The packets are being discarded.

Checking the Output Filter

Purpose Verify that Device R2's firewall filter is functioning properly.

Action From Device R2, enter the **show firewall filter log-discard** command.

```
user@R2> show firewall filter log-discard
Filter: log-discard
Counters:
Name                                                Bytes              Packets
counter                                               336                    4
```

Meaning As expected, the counter is being incremented.

Checking the Community Attribute

Purpose Verify that the route is being tagged with the community attribute.

Action From Device R1, enter the **show route extensive** command, using the neighbor address for Device R2, 192.0.2.101.

```
user@R1> show route 192.0.2.101 extensive

inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
192.0.2.101/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.0.2.101/32 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 684
                Address: 0x94141d8
                Next-hop reference count: 2
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                Session Id: 0x8000a
                State: <Active Ext>
                Local AS:   100 Peer AS:   200
                Age: 53:03
                Validation State: unverified
                Task: BGP_200.10.0.0.2+63097
                Announcement bits (1): 2-KRT
                AS path: 200 I
                Communities: 100:5555
                Accepted
                Localpref: 100
                Router ID: 192.168.0.2
```

Meaning As expected, when Device R2 advertises the 192.0.2.101 route to Device R1, Device R2 adds the 100:5555 community tag.

Related Documentation

- Understanding Forwarding Packets to the Discard Interface on page 103
- Example: Configuring Routing Policy Prefix Lists on page 261

CHAPTER 28

Routing Policy Configuration Statements

address-family

Syntax

```
address-family {
  inet {
    address;
    table table-name;
  }
  ccc {
    interface-name;
    standby;
    peer-unit unit-number;
    table table-name;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists], [edit policy-options condition if-route-exists],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify that the route must correspond to certain prefix type to be considered a match.

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

Related Documentation

- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*

aigp-originate

Syntax aigp-originate *distance*;

Hierarchy Level
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* term *term-name* then],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* then],
[edit policy-options policy-statement *policy-name* term *term-name* then],
[edit policy-options policy-statement *policy-name* then]

Release Information Statement introduced in Junos OS Release 12.1.

Description Originate an accumulated interior gateway protocol (AIGP) BGP attribute for a given prefix by export policy, using the **aigp-originate** policy action.

To originate an AIGP attribure, you need configure the policy action on only one node. The AIGP attribute is readvertised if the neighbors are AIGP enabled with the **aigp** statement in the BGP configuration.

Default If you omit the **aigp-originate** policy action, the node still readvertises the AIGP BGP attribute if AIGP is enabled in the BGP configuration. However, the node does not originate its own AIGP attribute for local prefixes.

As the route is readvertised by downstream nodes, the cost of the AIGP attribute reflects the IGP distance to the prefix (zero + IGP distance or configured distance + IGP distance).

Options ***distance***—(Optional) Associate an initial cost when advertising a local prefix with the AIGP BGP attribute.

Range: 0 through 4,294,967,295

Default: The initial cost associated with the AIGP attribute for a local prefix is zero. The ***distance*** option overrides the default zero value for the initial cost.

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

Related Documentation
- *Example: Configuring the Accumulated IGP Attribute for BGP*
- *aigp*

apply-path

Syntax apply-path *path*;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options prefix-list *name*],
[edit policy-options prefix-list *name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Expand a prefix list to include all prefixes pointed to by a defined path.

Options ***path***—String of elements composed of identifiers or configuration keywords that points to a set of prefixes. You can include wildcards (enclosed in angle brackets) to match more than one identifier. You cannot add a path element, including wildcards, after a leaf statement. Path elements, including wildcards, can only be used after a container statement.

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

Related Documentation

- Configuring Prefix Lists on page 46
- Example: Configuring Routing Policy Prefix Lists on page 261
- *Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List*

as-path (Policy Options)

Syntax as-path *name regular-expression*;

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.

Description Define an autonomous system (AS) path regular expression for use in a routing policy match condition.

Options ***name***—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 65,536 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

regular-expression—One or more regular expressions used to match the AS path.

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

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 57
- Example: Using AS Path Regular Expressions on page 283
- dynamic-db on page 456

as-path-group

Syntax

```
as-path-group group-name {
  as-path name regular-expression;
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for dynamic database configuration introduced in Junos OS Release 9.5.
Support for dynamic database configuration introduced in Junos OS Release 9.5 for EX Series switches.

Description

Define a group containing multiple AS path regular expressions for use in a routing policy match condition.

Options

group-name—Name that identifies the AS path group. One or more AS path regular expressions must be listed below the **as-path-group** hierarchy.

name—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

regular-expression—One or more regular expressions used to match the AS path.

Required Privilege Level

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

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 57
- dynamic-db on page 456

ccc (Routing Policy Condition)

Syntax

```
ccc {
  interface-name;
  standby;
  peer-unit unit-number;
  table table-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family],
[edit policy-options condition if-route-exists address-family],

Release Information

Statement introduced in Junos OS Release 13.2.

Description

Specify that the route must correspond to a CCC prefix to be considered a match.

Options

interface-name—Interface used to establish the CCC route.

The remaining statements are explained separately.

Required Privilege Level

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

Related Documentation

- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*

community (Policy Options)

Syntax

```
community name {
  invert-match;
  members [ community-ids ];
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.

Description

Define a community or extended community for use in a routing policy match condition.

Options

name—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters. To include spaces in the name, enclose it in quotation marks (“ ”).

invert-match—Invert the results of the community expression matching. The **community** match condition defines a regular expression and if it matches the community attribute of the received prefix, Junos OS returns a TRUE result. If not, Junos OS returns a FALSE result. The **invert-match** statement makes Junos OS behave to the contrary. If there is a match, Junos OS returns a FALSE result. If there is no match, Junos OS returns a TRUE result.

members *community-ids*—One or more community members. If you specify more than one member, you must enclose all members in brackets.

The format for ***community-ids*** is:

as-number:*community-value*

as-number is the AS number and can be a value in the range from 0 through 65,535. ***community-value*** is the community identifier and can be a number in the range from 0 through 65,535.

You also can specify ***community-ids*** for communities as one of the following well-known community names, which are defined in RFC 1997, *BGP Communities Attribute*:

- **no-export**—Routes containing this community name are not advertised outside a BGP confederation boundary.
- **no-advertise**—Routes containing this community name are not advertised to other BGP peers.
- **no-export-subconfed**—Routes containing this community name are not advertised to external BGP peers, including peers in other members' ASs inside a BGP confederation.

You can explicitly exclude BGP community information with a static route using the **none** option. Include **none** when configuring an individual route in the **route** portion of the **static** statement to override a **community** option specified in the **defaults** portion of the statement.

The format for extended ***community-ids*** is the following:

type:*administrator*:*assigned-number*

type is the type of extended community and can be either a **bandwidth**, **target**, **origin**, **domain-id**, **src-as**, or **rt-import** community or a 16-bit number that identifies a specific BGP extended community. The **target** community identifies the destination to which the route is going. The **origin** community identifies where the route originated. The **domain-id** community identifies the OSPF domain from which the route originated. The **src-as** community identifies the autonomous system from which the route originated. The **rt-import** community identifies the route to install in the routing table.

NOTE: For **src-as**, you can specify only an AS number and not an IP address. For **rt-import**, you can specify only an IP address and not an AS number.

administrator is the administrator. It is either an AS number or an IPv4 address prefix, depending on the type of extended community.

assigned-number identifies the local provider.

The format for linking a bandwidth with an AS number is:

bandwidth:*as-number*:*bandwidth*

as-number specifies the AS number and ***bandwidth*** specifies the bandwidth in bytes per second.

NOTE: In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295. To configure a **target** or **origin** extended community that includes a 4-byte AS number in the plain-number format, append the letter "L" to the end of number. For example, a target community with the 4-byte AS number 334,324 and an assigned number of 132 is represented as **target:334324L:132**.

In Junos OS Release 9.2 and later, you can also use AS-dot notation when defining a 4-byte AS number for the **target** and **origin** extended communities. Specify two integers 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 represented in plain-number format as 65546 is represented in AS-dot notation as 1.10.

Required Privilege Level

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

Related Documentation

- Understanding BGP Communities and Extended Communities as Routing Policy Match Conditions on page 67
- Understanding How to Define BGP Communities and Extended Communities on page 68
- dynamic-db on page 456

condition

Syntax

```
condition condition-name {
  dynamic-db;
  if-route-exists{
    address;
    address-family {
      inet {
        address;
        table table-name;
      }
      ccc {
        interface-name;
        standby;
        peer-unit unit-number;
        table table-name;
      }
    }
    table table-name;
  }
}
```

Hierarchy Level

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

Release Information

Statement introduced in Junos OS Release 9.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the address families introduced in Junos OS Release 13.2.

Description

Define a policy condition based on the existence of routes in specific tables for use in BGP export policies.

Options

condition-name—Name of the condition.

The remaining statements are explained separately.

Required Privilege Level

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

Related Documentation

- Understanding Conditional Installation of Prefixes in a Routing Table on page 92
- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*
- dynamic-db on page 456

damping (Policy Options)

Syntax

```
damping name {
    disable;
    half-life minutes;
    max-suppress minutes;
    reuse number;
    suppress number;
}
```

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define route flap damping properties to set on BGP routes.

Options

disable—Disable damping on a per-prefix basis. Any damping state that is present in the routing table for a prefix is deleted if damping is disabled.

half-life *minutes*—Decay half-life. ***minutes*** is the interval after which the accumulated figure-of-merit value is reduced by half if the route remains stable.

Range: 1 through 45

Default: 15 minutes

NOTE: **For the half-life, configure a value that is less than the max-suppress. If you do not, the configuration is rejected.**

max-suppress *minutes*—Maximum hold-down time. ***minutes*** is the maximum time that a route can be suppressed no matter how unstable it has been.

Range: 1 through 720

Default: 60 minutes

NOTE: **For the max-suppress, configure a value that is greater than the half-life. If you do not, the configuration is rejected.**

name—Name that identifies the set of damping parameters. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

reuse *number*—Reuse threshold. ***number*** is the figure-of-merit value below which a suppressed route can be used again.

Range: 1 through 20,000

Default: 750 (unitless)

suppress *number*—Cutoff (suppression) threshold. ***number*** is the figure-of-merit value above which a route is suppressed for use or inclusion in advertisements.

Range: 1 through 20,000

Default: 3000 (unitless)

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

Related Documentation
- Configuring BGP Flap Damping Parameters on page 83
- Example: Configuring Damping Parameters on page 375
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 384

dynamic-db

Syntax
dynamic-db;

Hierarchy Level
[edit logical-systems *logical-system-name* policy-options as-path *path-name*],
[edit logical-systems *logical-system-name* policy-options as-path-group *group-name*],
[edit logical-systems *logical-system-name* policy-options community *community-name*],
[edit logical-systems *logical-system-name* policy-options condition *condition-name*],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-statement-name*],
[edit logical-systems *logical-system-name* policy-options prefix-list *prefix-list-name*],
[edit policy-options as-path *path-name*],
[edit policy-options as-path-group *group-name*],
[edit policy-options community *community-name*],
[edit policy-options condition *condition-name*],
[edit policy-options policy-statement *policy-statement-name*],
[edit policy-options prefix-list *prefix-list-name*]

Release Information
Statement introduced in Junos OS Release 9.5.
Statement introduced in Junos OS Release 9.5 for EX Series switches.

Description
Define routing policies and policy objects that reference policies configured in the dynamic database at the **[edit dynamic]** hierarchy level.

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

Related Documentation
- Example: Configuring Dynamic Routing Policies on page 421

if-route-exists

Syntax

```
if-route-exists {
  address;
  address-family {
    inet {
      address;
      table table-name;
    }
    ccc {
      interface-name;
      standby;
      peer-unit unit-number;
      table table-name;
    }
  }
  table table-name;
}
```

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

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify the route match conditions.

Options (Optional) ***address***—Specify the IP address that the route must have to be considered a match.

The remaining statements are explained separately.

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

Related Documentation

- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*
- Example: Configuring Conditional Installation of Prefixes in a Routing Table on page 405

export (Protocols BGP)

Syntax export [*policy-names*];

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

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more policies to routes being exported from the routing table into BGP.

If you specify more than one policy, they are evaluated in the order specified, from left to right, 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. If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a chain sets a route's metric to 500, this route matches the criterion of **metric 500** defined in the next policy.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Understanding Route Advertisement*
- *Routing Policy Feature Guide for Routing Devices*
- import on page 467

export (Protocols IS-IS)

Syntax export [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 12.1 for the QFX Series.

Description Apply one or more policies to routes being exported from the routing table into IS-IS.

All routing protocols store the routes that they learn in the routing table. The routing table uses this collected route information to determine the active routes to destinations. The routing table then installs the active routes into its forwarding table and exports them into the routing protocols. It is these exported routes that the protocols advertise.

For each protocol, you control which routes the protocol stores in the routing table and which routes the routing table exports into the protocol from the routing table by defining a *routing policy* for that protocol.

NOTE: For IS-IS, you cannot apply routing policies that affect how routes are imported into the routing table; doing so with a link-state protocol can easily lead to an inconsistent topology database.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation

- *Example: Redistributing OSPF Routes into IS-IS*
- *Example: Configuring an IS-IS Default Route Policy on Logical Systems*

export (Protocols DVMRP)

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols dvmrp],
[edit protocols dvmrp]

Release Information Statement introduced before Junos OS Release 7.4.

Description Apply one or more policies to routes being exported from the routing table into DVMRP. If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching policy is applied to the route. If no match is found, the routing table exports into DVMRP only the routes that it learned from DVMRP and direct routes.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- import on page 468
- *Example: Configuring DVMRP to Announce Unicast Routes*

export (Protocols LDP)

Syntax export [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.

Description Apply policy filters to outbound LDP label bindings. Filters are applied to all label bindings from all neighbors.

Options ***policy-names***—Name of one or more routing policies.

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

Related Documentation
- *Filtering Outbound LDP Label Bindings*

export (Protocols MSDP)

Syntax export [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols msdp],
[edit logical-systems *logical-system-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* protocols msdp peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp peer *address*],
[edit protocols msdp],
[edit protocols msdp group *group-name*],
[edit protocols msdp group *group-name* peer *address*],
[edit protocols msdp peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp peer *address*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.1 for the QFX Series.

Description Apply one or more policies to routes being exported from the routing table into MSDP.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Example: Configuring MSDP in a Routing Instance*
- import on page 470

export (Protocols OSPF)

Syntax export [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit protocols (ospf | ospf3)],
[edit protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for the **realm** statement introduced in Junos OS Release 9.2.
Support for the **realm** statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more policies to routes being exported from the routing table into OSPF.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation

- *Understanding OSPF Routing Policy*
- *Import and Export Policies for Network Summaries Overview*
- import on page 471
- *Routing Policy Feature Guide for Routing Devices*

export (Protocols PIM)

Syntax export [*policy-names*];

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

Release Information
Statement introduced in Junos OS Release 9.6.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more export policies to control outgoing PIM join and prune messages. PIM join and prune filters can be applied to PIM-SM and PIM-SSM messages. PIM join and prune filters cannot be applied to PIM-DM messages.

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

Related Documentation
- *Filtering Outgoing PIM Join Messages*

export (Protocols PIM Bootstrap)

Syntax export [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols pim rp bootstrap family (inet | inet6)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim rp bootstrap family (inet | inet6)],
[edit protocols pim rp bootstrap family (inet | inet6)],
[edit routing-instances *routing-instance-name* protocols pim rp bootstrap family (inet | inet6)]

Release Information
Statement introduced in Junos OS Release 7.6.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more export policies to control outgoing PIM bootstrap messages.

Options ***policy-names***—Name of one or more import policies.

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

Related Documentation
- *Configuring PIM Bootstrap Properties for IPv4*
- *Configuring PIM Bootstrap Properties for IPv4 or IPv6*
- import (Protocols PIM Bootstrap) on page 473

export (Protocols RIP)

Syntax export [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Apply a policy to routes being exported to the neighbors.

By default, RIP does not export routes it has learned to its neighbors. To enable RIP to export routes, apply one or more export policies.

If no routes match the policies, the local routing device does not export any routes to its neighbors. Export policies override any metric values determined through calculations involving the values configured with the **metric-in** and **metric-out** statements.

NOTE: The export policy on RIP does not support manipulating routing information of the next hop.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation

- *Example: Configuring RIP*
- import on page 474

export (Protocols RIPng)

Syntax export [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for routing instances introduced in Junos OS Release 9.0.

Description Apply a policy or list of policies to routes being exported to the neighbors.

By default, RIPng does not export routes it has learned to its neighbors. To have RIPng export routes, apply one or more export policies. To apply export policies and to filter routes being exported from the local routing device to its neighbors, include the **export** statement and list the name of the policy to be evaluated.

You can define one or more export policies. If no routes match the policies, the local routing device does not export any routes to its neighbors. Export policies override any metric values determined through calculations involving the values configured with the **metric-in** and **metric-out** statements.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Example: Configuring RIPng*
- import on page 475

export (Routing Options)

Syntax export [*policy-name*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.

Description Apply one or more policies to routes being exported from the routing table into the forwarding table.

In the **export** statement, list the name of the routing policy to be evaluated when routes are being exported from the routing table into the forwarding table. Only active routes are exported from the routing table.

You can reference the same routing policy one or more times in the same or a different **export** statement.

You can apply export policies to routes being exported from the routing table into the forwarding table for the following features:

- Per-packet load balancing
- Class of service (CoS)

Options ***policy-name***—Name of one or more policies.

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

Related Documentation

- *Example: Load Balancing BGP Traffic*
- *Routing Policy Feature Guide for Routing Devices*
- How a Routing Policy Is Evaluated on page 17

import (Protocols BGP)

Syntax import [*policy-names*];

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

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more routing policies to routes being imported into the Junos OS routing table from BGP.

If you specify more than one policy, they are evaluated in the order specified, from left to right, 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. The policy framework software evaluates the routing policies in a chain sequentially. If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a chain sets a route's metric to 500, this route matches the criterion of **metric 500** defined in the next policy.

It is also 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, all routes received are discarded. When this statement is configured and the inbound policy changes, Junos OS re-advertises all the routes advertised by the peer.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Example: Configuring BGP Interactions with IGPs*
- *Understanding Route Advertisement*
- *Routing Policy Feature Guide for Routing Devices*
- export on page 458

import (Protocols DVMRP)

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols dvmrp],
[edit protocols dvmrp]

Release Information Statement introduced before Junos OS Release 7.4.

Description Apply one or more policies to routes being imported into the routing table from DVMRP. If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching policy is applied to the route. If no match is found, DVMRP shares with the routing table only those routes that were learned from DVMRP routers.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- export on page 460
- *Example: Configuring DVMRP to Announce Unicast Routes*

import (Protocols LDP)

Syntax import [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.

Description Apply policy filters to received LDP label bindings. Filters are applied to all label bindings from all neighbors.

Options ***policy-names***—Name of one or more routing policies.

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

Related Documentation
- *Filtering Inbound LDP Label Bindings*

import (Protocols MSDP)

Syntax import [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols msdp],
[edit logical-systems *logical-system-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* protocols msdp peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp peer *address*],
[edit protocols msdp],
[edit protocols msdp group *group-name*],
[edit protocols msdp group *group-name* peer *address*],
[edit protocols msdp peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp peer *address*]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.1 for the QFX Series.

Description Apply one or more policies to routes being imported into the routing table from MSDP.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Example: Configuring MSDP in a Routing Instance*
- export on page 461

import (Protocols OSPF)

Syntax import [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit protocols (ospf | ospf3)],
[edit protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for the **realm** statement introduced in Junos OS Release 9.2.
Support for the **realm** statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Filter OSPF routes from being added to the routing table.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation

- *Understanding OSPF Routing Policy*
- *Import and Export Policies for Network Summaries Overview*
- export on page 462
- *Routing Policy Feature Guide for Routing Devices*

import (Protocols PIM)

Syntax import [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more policies to routes being imported into the routing table from PIM. Use the **import** statement to filter PIM join messages and prevent them from entering the network.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation

- *Filtering Incoming PIM Join Messages*

import (Protocols PIM Bootstrap)

Syntax import [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols pim rp bootstrap (inet | inet6)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim rp bootstrap (inet | inet6)],
[edit protocols pim rp bootstrap (inet | inet6)],
[edit routing-instances *routing-instance-name* protocols pim rp bootstrap (inet | inet6)]

Release Information
Statement introduced in Junos OS Release 7.6.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Apply one or more import policies to control incoming PIM bootstrap messages.

Options ***policy-names***—Name of one or more import policies.

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

Related Documentation
- *Configuring PIM Bootstrap Properties for IPv4*
- *Configuring PIM Bootstrap Properties for IPv4 or IPv6*
- export (Protocols PIM Bootstrap) on page 463

import (Protocols RIP)

Syntax import [*policy-names*];

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

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 12.1 for the QFX Series.

Description Apply one or more policies to routes being imported by the local routing device from neighbors.

Options ***policy-names***—Name of one or more policies.

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

Related Documentation
- *Example: Applying Policies to RIP Routes Imported from Neighbors*
- *Routing Policy Feature Guide for Routing Devices*
- export on page 464

import (Protocols RIPng)

Syntax

import [*policy-names*];

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for routing instances introduced in Junos OS Release 9.0.

Description

Apply one or more policies to routes being imported into the local routing device from its neighbors.

Options

policy-names—Name of one or more policies.

Required Privilege Level

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

Related Documentation

- *Example: Applying Policies to RIPng Routes Imported from Neighbors*
- export on page 465

import (Routing Options)

Syntax import [*policy-names*];

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

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.

Description Specify one or more import policies to use for route resolution.

Options ***policy-names***—Name of one or more import policies.

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

Related Documentation
- *Example: Configuring Route Resolution on PE Routers*

inet (Routing Policy Condition)

Syntax
```
inet {
  address;
  table table-name;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family],
[edit policy-options condition if-route-exists address-family],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify that the route must correspond to a IPv4 prefix to be considered a match.

Options (Optional) ***address***—Specify the IP address that the route must have to be considered a match.

The remaining statements are explained separately.

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

Related Documentation
- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*

peer-unit (Routing Policy Condition)

Syntax peer-unit *unit-number*;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family ccc],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify the associated logical tunnel interface's peer-unit. This is required for logical-tunnel-based routes.

Options ***unit-number***—Logical unit number of the logical tunnel peer interface.

Range: 0 through 8192

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

Related Documentation

- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*

policy-options

Syntax

```
policy-options {
  as-path name regular-expression;
  as-path-group group-name;
  community name {
    invert-match;
    members [ community-ids ];
  }
  condition condition-name {
    if-route-exists address table table-name;
  }
  damping name {
    disable;
    half-life minutes;
    max-suppress minutes;
    reuse number;
    suppress number;
  }
  policy-statement policy-name {
    term term-name {
      from {
        family;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
        policy subroutine-policy-name;
      }
      then actions;
      default-action (accept | reject);
    }
  }
  prefix-list name {
    ip-addresses;
  }
}
```

Hierarchy Level

[edit],
[edit dynamic],
[edit dynamic-profiles *profile-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure routing policy.

The statements are explained separately.

Required Privilege Level

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

Related Documentation

- Example: Using Routing Policy in an ISP Network on page 149

policy-statement

Syntax

```
policy-statement policy-name {
  term term-name {
    from {
      family family-name;
      match-conditions;
      policy subroutine-policy-name;
      prefix-list prefix-list-name;
      prefix-list-filter prefix-list-name match-type <actions>;
      route-filter destination-prefix match-type <actions>;
      source-address-filter source-prefix match-type <actions>;
    }
    to {
      match-conditions;
      policy subroutine-policy-name;
    }
    then actions;
  }
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
inet-mdt option introduced in Junos OS Release 10.0R2.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
route-target option introduced in Junos OS Release 12.2.

Description

Define a routing policy, including subroutine policies.

A *term* is a named structure in which match conditions and actions are defined. Routing policies are made up of one or more terms. Each routing policy term is identified by a term 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 term contains a set of match conditions and a set of actions:

- Match conditions are criteria that a route must match before the actions can be applied. If a route matches all criteria, one or more actions are applied to the route.
- Actions specify whether to accept or reject the route, control how a series of policies are evaluated, and manipulate the characteristics associated with a route.

Generally, a router compares a route against the match conditions of each term in a routing policy, starting with the first and moving through the terms in the order in which they are defined, until a match is made and an explicitly configured or default action of

accept or **reject** is taken. If none of the terms in the policy match the route, the router compares the route against the next policy, and so on, until either an action is taken or the default policy is evaluated.

If none of the match conditions of each term evaluates to true, the final action is executed. The final action is defined in an unnamed term. Additionally, you can define a default action (either **accept** or **reject**) that overrides any action intrinsic to the protocol.

The order of match conditions in a term is not relevant, because a route must match all match conditions in a term for an action to be taken.

To list the routing policies under the **[edit policy-options]** hierarchy level by **policy-statement** ***policy-name*** in alphabetical order, enter the **show policy-options** configuration command.

Options

actions—(Optional) One or more actions to take if the conditions match. The actions are described in "Configuring Flow Control Actions" on page 120.

family *family-name*—(Optional) Specify an address family protocol. Specify **inet** for IPv4. Specify **inet6** for 128-bit IPv6, and to enable interpretation of IPv6 router filter addresses. For IS-IS traffic, specify **iso**. For IPv4 multicast VPN traffic, specify **inet-mvpn**. For IPv6 multicast VPN traffic, specify **inet6-mvpn**. For multicast-distribution-tree (MDT) IPv4 traffic, specify **inet-mdt**. For BGP route target VPN traffic, specify**route-target**.

NOTE: When **family** is not specified, the routing device or routing instance uses the address family or families carried by BGP. If multiprotocol BGP (MP-BGP) is enabled, the policy defaults to the protocol family or families carried in the network layer reachability information (NLRI) as configured in the *family* statement for BGP. If MP-BGP is not enabled, the policy uses the default BGP address family unicast IPv4.

from—(Optional) Match a route based on its source address.

match-conditions—(Optional in **from** statement; required in **to** statement) One or more conditions to use to make a match. The qualifiers are described in "Routing Policy Match Conditions" on page 107.

policy *subroutine-policy-name*—Use another policy as a match condition within this policy. The name identifying the subroutine policy can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" "). Policy names cannot take the form **__.*-internal__**, as this form is reserved. For information about how to configure subroutines, see "Understanding Policy Subroutines in Routing Policy Match Conditions" on page 51.

policy-name—Name that identifies the policy. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" ").

prefix-list *prefix-list-name* —Name of a list of IPv4 or IPv6 prefixes.

prefix-list-filter *prefix-list-name*—Name of a prefix list to evaluate using qualifiers; ***match-type*** is the type of match (see "Configuring Prefix List Filters" on page 25), and ***actions*** is the action to take if the prefixes match.

route-filter *destination-prefix match-type <actions>*—(Optional) List of routes on which to perform an immediate match; ***destination-prefix*** is the IPv4 or IPv6 route prefix to match, ***match-type*** is the type of match (see "Configuring Route Filters" on page 27), and ***actions*** is the action to take if the ***destination-prefix*** matches.

source-address-filter *source-prefix match-type <actions>*—(Optional) Unicast source addresses in multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments on which to perform an immediate match. ***source-prefix*** is the IPv4 or IPv6 route prefix to match, ***match-type*** is the type of match (see

"Configuring Route Filters" on page 27), and ***actions*** is the action to take if the ***source-prefix*** matches.

term ***term-name***—Name that identifies the term. The term name must be unique in the policy. It can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (" "). A policy statement can include multiple terms. We recommend that you name all terms. However, you do have the option to include an unnamed term which must be the final term in the policy. To configure an unnamed term, omit the **term** statement when defining match conditions and actions.

to—(Optional) Match a route based on its destination address or the protocols into which the route is being advertised.

then—(Optional) Actions to take on matching routes. The actions are described in "Configuring Flow Control Actions" on page 120 and "Configuring Actions That Manipulate Route Characteristics" on page 121.

Required Privilege Level

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

Related Documentation

- dynamic-db on page 456

prefix-list

Syntax

```
prefix-list name {
  ip-addresses;
  apply-path path;
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the **vpls** protocol family introduced in Junos OS Release 10.2.

Description

Define a list of IPv4 or IPv6 address prefixes for use in a routing policy statement or firewall filter statement.

You can configure up to 85,325 prefixes in each prefix list. To configure more than 85,325 prefixes, configure multiple prefix lists and apply them to multiple firewall filter terms.

Options

name—Name that identifies the list of IPv4or IPv6 address prefixes.

ip-addresses—List of IPv4 or IPv6 address prefixes, one IP address per line in the configuration.

The remaining statement is explained separately.

Required Privilege Level

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

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 45
- *Firewall Filter Match Conditions Based on Address Fields*
- Example: Configuring Routing Policy Prefix Lists on page 261
- *Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List*

prefix-list-filter

Syntax prefix-list-filter *prefix-list-name match-type* <*actions*>;

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

Release Information Statement introduced before Junos OS Release 7.4.

Description Evaluate a list of prefixes within a prefix list using specified qualifiers.

Options ***prefix-list-name***—Name of the prefix list to evaluate.

match-type—Prefix length qualifiers.

actions—(Optional) Actions to take on match.

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

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 45
- Example: Configuring Routing Policy Prefix Lists on page 261

rtf-prefix-list

Syntax rtf-prefix-list *name route-targets*

Hierarchy Level [edit logical-systems *logical-system-name* policy-options],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* term *term-name*],
[edit policy-options],
[edit policy-options policy-statement *policy-name* term *term-name*]

Release Information Statement introduced in Junos OS Release 12.2.

Description Define a list of route target prefixes for use in a routing policy statement. These prefixes are only useful for filtering routes in the bpg.rtarget.0 table.

The route target filtering prefix is in the format: *AS number:route target extended community/length*. The first number represents the autonomous system (AS) of the device that sent the advertisement. The second group of numbers represent the route target extended community. The format of the extended community is the same as the extended community type **target:**. For more information about extended communities, see "Understanding How to Define BGP Communities and Extended Communities" on page 68.

In this route target prefix example 200:200:101/96, 200 is the AS number, 200:101 is the BGP extended community used for the route target, and 96 is the prefix length.

For more information about the route target community, see RFC 4360, *BGP Extended Communities Attribute*.

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

Options ***name***—Name that identifies the list of route target filtering prefixes. 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 quotation marks (" ").

route-targets—List of route target filtering prefixes, one route target filter per line in the configuration. When you use the **rtf-prefix-list** statement as a match condition, you do not have the option of configuring the list of route target filtering prefixes. You must first define and configure the route target filtering prefixes with the policy-options statement.

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

Related Documentation

- *Example: Configuring an Export Policy for BGP Route Target Filtering*
- *Configuring BGP Route Target Filtering for VPNs*
- *Understanding Proxy BGP Route Target Filtering*

- Understanding How to Define BGP Communities and Extended Communities on page 68 in the *Routing Policy Feature Guide for Routing Devices*
- *family route-target*

standby (Routing Policy Condition)

Syntax standby;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family ccc],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify that the route must be in standby state to be considered a match.

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

Related Documentation
- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*

table

Syntax table *table-name*;

Hierarchy Level
[edit dynamic policy-options condition],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family inet],
[edit policy-options condition if-route-exists],
[edit policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family inet]

Release Information
Statement introduced in Junos OS Release 9.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the address families introduced in Junos OS Release 13.2.

Description Specify a routing table in which the route must exist for the condition to be met and the route to be considered a match.

Options **table *table-name***—Routing table name, such as inet.0.

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

Related Documentation

- Understanding Conditional Installation of Prefixes in a Routing Table on page 92
- *Example: Configuring Pseudowire Redundancy for Mobile Backhaul Scenarios*
- dynamic-db on page 456

PART 3

Administration

- Routing Policy Operational Commands on page 491

CHAPTER 29

Routing Policy Operational Commands

show policy

Syntax
show policy
<logical-system (all | *logical-system-name*)>
<*policy-name*>

Syntax (EX Series Switches)
show policy
<*policy-name*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display information about configured routing policies.

Options
none—List the names of all configured routing policies.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

policy-name—(Optional) Show the contents of the specified policy.

Required Privilege Level
view

Related Documentation
- show policy damping on page 496

List of Sample Output
show policy on page 492
show policy policy-name on page 493
show policy (Multicast Scoping) on page 493

Output Fields
Table 26 on page 492 lists the output fields for the **show policy** command. Output fields are listed in the approximate order in which they appear.

Table 26: show policy Output Fields

Field Name	Field Description
policy-name	Name of the policy listed.
term	Policy term listed.
from	Match condition for the policy.
then	Action for the policy.

Sample Output

show policy

```
user@host> show policy
```

```
Configured policies:
__vrf-export-red-internal__
__vrf-import-red-internal__
red-export
all_routes
```

show policy policy-name

```
user@host> show policy test-statics
Policy test-statics:
    from
         3.0.0.0/8  accept
         3.1.0.0/16  accept
    then reject
```

show policy (Multicast Scoping)

```
user@host> show policy test-statics
Policy test-statics:
    from
         multicast-scoping == 8
```

show policy conditions

Syntax

show policy conditions
<*condition-name*>
<detail>
<dynamic>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show policy conditions
<*condition-name*>
<detail>
<dynamic>

Release Information

Command introduced in Junos OS Release 9.0.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display all the configured conditions as well as the routing tables with which the configuration manager is interacting. If the **detail** keyword is included, the output also displays dependent routes for each condition.

Options

none—Display all configured conditions and associated routing tables.

condition-name—(Optional) Display information about the specified condition only.

detail—(Optional) Display the specified level of output.

dynamic—(Optional) Display information about the conditions in the dynamic database.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show policy conditions detail on page 495

Output Fields

Table 27 on page 494 lists the output fields for the **show policy conditions** command. Output fields are listed in the approximate order in which they appear.

Table 27: show policy conditions Output Fields

Field Name	Field Description	Level of Output
Condition	Name of configured condition.	All levels
event	Condition type. If the **if-route-exists** option is configured, the event type is: **Existence of a route in a specific routing table**.	All levels
Dependent routes	List of routes dependent on the condition, along with the latest generation number.	**detail**
Condition tables	List of routing tables associated with the condition, along with the latest generation number and number of dependencies.	All levels

Table 27: show policy conditions Output Fields *(continued)*

Field Name	Field Description	Level of Output
If-route-exists conditions	List of conditions configured to look for a route in the specified table.	All levels

Sample Output

show policy conditions detail

```
user@host> show policy conditions detail
Configured conditions:
Condition primary (static), event: Existence of a route in a specific routing
table
Dependent routes:
 8.41.0.0/24, generation 18

Condition standby (static), event: Existence of a route in a specific routing
table
Dependent routes:
 8.41.0.0/24, generation 18

Condition tables:
Table mpls.0, generation 0, dependencies 0, If-route-exists conditions: primary
(static) standby (static)
Table l3vpn.inet.0, generation 633, dependencies 2
```

show policy damping

Syntax show policy damping
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switch and QFX Series) show policy damping

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.

Description Display information about BGP route flap damping parameters.

Options **none**—Display information about BGP route flap damping parameters.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information In the output from this command, figure-of-merit values correlate with the probability of future instability of a routing device. Routes with higher figure-of-merit values are suppressed for longer periods of time. The figure-of-merit value decays exponentially over time. A figure-of-merit value of zero is assigned to each new route. The value is increased each time the route is withdrawn or readvertised, or when one of its path attributes changes.

Required Privilege Level view

Related Documentation
- "Configuring BGP Flap Damping Parameters" in the *Routing Policy Feature Guide for Routing Devices*
- *clear bgp damping*
- show route damping on page 526

List of Sample Output show policy damping on page 497

Output Fields Table 28 on page 496 describes the output fields for the **show policy damping** command. Output fields are listed in the approximate order in which they appear.

Table 28: show policy damping Output Fields

Field Name	Field Description
Halflife	Decay half-life, in minutes. The value represents the period during which the accumulated figure-of-merit value is reduced by half if the route remains stable. If a route has flapped, but then becomes stable, the figure-of-merit value for the route decays exponentially. For example, for a route with a figure-of-merit value of 1500, if no incidents occur, its figure-of-merit value is reduced to 750 after 15 minutes and to 375 after another 15 minutes.

Table 28: show policy damping Output Fields *(continued)*

Field Name	Field Description
Reuse merit	Figure-of-merit value below which a suppressed route can be used again. A suppressed route becomes reusable when its figure-of-merit value decays to a value below a reuse threshold, and the route once again is considered usable and can be installed in the forwarding table and exported from the routing table.
Suppress/cutoff merit	Figure-of-merit value above which a route is suppressed for use or inclusion in advertisements. When a route's figure-of-merit value reaches a particular level, called the cutoff or suppression threshold, the route is suppressed. When a route is suppressed, the routing table no longer installs the route into the forwarding table and no longer exports this route to any of the routing protocols.
Maximum suppress time	Maximum hold-down time, in minutes. The value represents the maximum time that a route can be suppressed no matter how unstable it has been before this period of stability.
Computed values	• **Merit ceiling**—Maximum merit that a flapping route can collect. • **Maximum decay**—Maximum decay half-life, in minutes.

Sample Output

show policy damping

```
user@host> 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 "standard-damping":
  Halflife: 10 minutes
  Reuse merit: 4000 Suppress/cutoff merit: 8000
  Maximum suppress time: 30 minutes
  Computed values:
    Merit ceiling: 32120
    Maximum decay: 12453
```

show route

Syntax

show route
<all>
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>
<private>

Syntax (EX Series Switches)

show route
<all>
<*destination-prefix*>
<private>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Option **private** introduced in Junos OS Release 9.5.
Option **private** introduced in Junos OS Release 9.5 for EX Series switches.

Description

Display the active entries in the routing tables.

Options

none—Display brief information about all active entries in the routing tables.

all—(Optional) Display information about all routing tables, including private, or internal, routing tables.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

private—(Optional) Display information only about all private, or internal, routing tables.

Required Privilege Level

view

Related Documentation

- *Example: Configuring RIP*
- *Example: Configuring RIPng*
- *Example: Configuring IS-IS*
- *Examples: Configuring Internal BGP Peering*
- *Examples: Configuring External BGP Peering*
- *Examples: Configuring OSPF Routing Policy*

List of Sample Output

show route on page 501
show route on page 501
show route destination-prefix on page 502
show route extensive on page 502

Output Fields Table 29 on page 499 describes the output fields for the **show route** command. Output fields are listed in the approximate order in which they appear.

Table 29: show route Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active). • **holddown** (routes that are in the pending state before being declared inactive). A holddown route was once the active route and is no longer the active route. The route is in the holddown state because a protocol still has interest in the route, meaning that the interest bit is set. A protocol might have its interest bit set on the previously active route because the protocol is still advertising the route. The route will be deleted after all protocols withdraw their advertisement of the route and remove their interest bit. A persistent holddown state often means that the interested protocol is not releasing its interest bit properly. However, if you have configured advertisement of multiple routes (with the **add-path** or **advertise-inactive** statement), the holddown bit is most likely set because BGP is advertising the route as an active route. In this case, you can ignore the holddown state because nothing is wrong. • **hidden** (routes that are not used because of a routing policy).
destination-prefix	Route destination (for example:10.0.0.1/24). Sometimes the route information is presented in another format, such as: • *MPLS-label* (for example, 80001). • *interface-name* (for example, ge-1/0/2). • *neighbor-address*:*control-word-status*:*encapsulation type*:*vc-id*:*source* (Layer 2 circuit only. For example, 10.1.1.195:NoCtrlWord:1:1:Local/96): ◦ *neighbor-address*—Address of the neighbor. ◦ *control-word-status*—Whether the use of the control word has been negotiated for this virtual circuit: **NoCtrlWord** or **CtrlWord**. ◦ *encapsulation type*—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport. ◦ *vc-id*—Virtual circuit identifier. ◦ *source*—Source of the advertisement: **Local** or **Remote**.
[*protocol*, *preference* **]**	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.

Table 29: show route Output Fields *(continued)*

Field Name	Field Description
weeks:days hours:minutes:seconds	How long the route been known (for example, **2w4d 13:11:14**, or 2 weeks, 4 days, 13 hours, 11 minutes, and 14 seconds).
metric	Cost value of the indicated route. For routes within an AS, the cost is determined by the IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
localpref	Local preference value included in the route.
from	Interface from which the route was received.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
validation-state	(BGP-learned routes) Validation status of the route: • **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. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
to	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route. If the destination is **Discard**, traffic is dropped.

Table 29: show route Output Fields *(continued)*

Field Name	Field Description
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • **Weight**—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • **Balance**—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing. • ***lsp-path-name***—Name of the LSP used to reach the next hop. • ***label-action***—MPLS label and operation occurring at the next hop. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label). For VPNs, expect to see multiple **push** operations, corresponding to the inner and outer labels required for VPN routes (in the case of a direct PE-to-PE connection, the VPN route would have the inner label push only).

Sample Output

show route

```
user@host> show route
inet.0: 11 destinations, 12 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:65500:1:10.0.0.20/240
                   *[MVPN/70] 19:53:41, metric2 1
                      Indirect
1:65500:1:10.0.0.40/240
                   *[BGP/170] 19:53:29, localpref 100, from 10.0.0.30
                      AS path: I
                    > to 10.0.24.4 via lt-0/3/0.24, label-switched-path toD
                    [BGP/170] 19:53:26, localpref 100, from 10.0.0.33
                      AS path: I
                    > to 10.0.24.4 via lt-0/3/0.24, label-switched-path toD
1:65500:1:10.0.0.60/240
                   *[BGP/170] 19:53:29, localpref 100, from 10.0.0.30
                      AS path: I
                    > to 10.0.28.8 via lt-0/3/0.28, label-switched-path toF
                    [BGP/170] 19:53:25, localpref 100, from 10.0.0.33
                      AS path: I
                    > to 10.0.28.8 via lt-0/3/0.28, label-switched-path toF
```

show route

The following sample output shows a VPN route with composite next hops enabled. The first **Push** operation corresponds to the outer label. The second **Push** operation corresponds to the inner label.

```
user@host> show route 70.0.0.0

13979:665001.inet.0: 871 destinations, 3556 routes (871 active, 0 holddown, 0
hidden)
```

```
+ = Active Route, - = Last Active, * = Both

70.0.0.0/24        @[BGP/170] 00:28:32, localpref 100, from 10.9.9.160
                      AS path: 13980 ?
                    > to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
                    [BGP/170] 00:28:28, localpref 100, from 10.9.9.169
                      AS path: 13980 ?
                    > to 10.100.0.42 via ae2.0, Push 126016, Push 300368(top)
                    #[Multipath/255] 00:28:28, metric2 102
                    > to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
                      to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
```

show route destination-prefix

```
user@host> show route 172.16.0.0/12

inet.0: 10 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.0/12      *[Static/5] 2w4d 12:54:27
                    > to 192.168.167.254 via fxp0.0
```

show route extensive

```
user@host> show route extensive
v1.mvpn.0: 5 destinations, 8 routes (5 active, 1 holddown, 0 hidden)
1:65500:1:10.0.0.40/240 (1 entry, 1 announced)
    *BGP   Preference: 170/-101
          PMSI: Flags 0x0: Label[0:0:0]: PIM-SM: Sender 10.0.0.40 Group 225.1.1.1

           Next hop type: Indirect
           Address: 0x92455b8
           Next-hop reference count: 2
           Source: 10.0.0.30
           Protocol next hop: 10.0.0.40
           Indirect next hop: 2 no-forward
           State: <Active Int Ext>
                    Local AS: 65500 Peer AS: 65500
           Age: 3  Metric2: 1
           Task: BGP_65500.10.0.0.30+179
           Announcement bits (2): 0-PIM.v1 1-mvpn global task
           AS path: I (Originator) Cluster list:  10.0.0.30
           AS path:  Originator ID: 10.0.0.40
           Communities: target:65520:100
           Import Accepted
           Localpref: 100
           Router ID: 10.0.0.30
           Primary Routing Table bgp.mvpn.0
           Indirect next hops: 1
                Protocol next hop: 10.0.0.40 Metric: 1
                Indirect next hop: 2 no-forward
                Indirect path forwarding next hops: 1
                      Next hop type: Router
                      Next hop: 10.0.24.4 via lt-0/3/0.24 weight 0x1
                10.0.0.40/32 Originating RIB: inet.3
                      Metric: 1                       Node path count: 1
                      Forwarding nexthops: 1
                         Nexthop: 10.0.24.4 via lt-0/3/0.24
```

show route active-path

Syntax
show route active-path
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route active-path
<brief | detail | extensive | terse>

Release Information
Command introduced in Junos OS Release 8.0.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display all active routes for destinations. An active route is a route that is selected as the best path. Inactive routes are not displayed.

Options
none—Display all active routes.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route active-path on page 503
show route active-path brief on page 504
show route active-path detail on page 504
show route active-path extensive on page 505
show route active-path terse on page 507

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route active-path

```
user@host> show route active-path

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.70.19/32    *[Direct/0] 21:33:52
                    > via lo0.0
10.255.71.50/32    *[IS-IS/15] 00:18:13, metric 10
                    > to 100.1.2.1 via so-2/1/3.0
100.1.2.0/24       *[Direct/0] 00:18:36
                    > via so-2/1/3.0
100.1.2.2/32       *[Local/0] 00:18:41
                      Local via so-2/1/3.0
192.168.64.0/21    *[Direct/0] 21:33:52
```

```
                     > via fxp0.0
192.168.70.19/32   *[Local/0] 21:33:52
                       Local via fxp0.0
```

show route active-path brief

The output for the **show route active-path brief** command is identical to that for the **show route active-path** command. For sample output, see show route active-path on page 503.

show route active-path detail

```
user@host> show route active-path detail

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)

10.255.70.19/32 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via lo0.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

10.255.71.50/32 (1 entry, 1 announced)
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 397
                Next-hop reference count: 4
                Next hop: 100.1.2.1 via so-2/1/3.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:31      Metric: 10
                Task: IS-IS
                Announcement bits (4): 0-KRT 2-IS-IS 5-Resolve tree 2 6-Resolve
tree 3
                AS path: I

100.1.2.0/24 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via so-2/1/3.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:54
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

100.1.2.2/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: so-2/1/3.0
                State: ‹Active NoReadvrt Int›
```



```
                Local AS:   200
                Age: 21:59
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.64.0/21 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via fxp0.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.70.19/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: fxp0.0
                State: ‹Active NoReadvrt Int›
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I
```

show route active-path extensive

```
user@host> show route active-path extensive

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
10.255.70.19/32 (1 entry, 1 announced)
TSI:
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via lo0.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:39:47
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

10.255.71.50/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.255.71.50/32 -> {100.1.2.1}
IS-IS level 2, LSP fragment 0
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 397
                Next-hop reference count: 4
                Next hop: 100.1.2.1 via so-2/1/3.0, selected
                State: ‹Active Int›
```

```
                Local AS:   200
                Age: 24:08      Metric: 10
                Task: IS-IS
                Announcement bits (4): 0-KRT 2-IS-IS 5-Resolve tree 2 6-Resolve
tree 3
                AS path: I

100.1.2.0/24 (1 entry, 1 announced)
TSI:
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via so-2/1/3.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 24:31
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

100.1.2.2/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: so-2/1/3.0
                State: ‹Active NoReadvrt Int›
                Local AS:   200
                Age: 24:36
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.64.0/21 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via fxp0.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:39:47
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.70.19/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: fxp0.0
                State: ‹Active NoReadvrt Int›
                Local AS:   200
                Age: 21:39:47
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I
```


show route active-path terse

```
user@host> show route active-path terse

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.19/32    D   0                     >lo0.0
* 10.255.71.50/32    I  15         10          >100.1.2.1
* 100.1.2.0/24       D   0                     >so-2/1/3.0
* 100.1.2.2/32       L   0                      Local
* 192.168.64.0/21    D   0                     >fxp0.0
* 192.168.70.19/32   L   0                      Local
```

show route advertising-protocol

Syntax show route advertising-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Release Information Command introduced before Junos OS Release 7.4.

Description Display the routing information as it has been prepared for advertisement to a particular neighbor of a particular dynamic routing protocol.

Options **brief | detail | extensive | terse**—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

neighbor-address—Address of the neighboring router to which the route entry is being transmitted.

protocol—Protocol transmitting the route:

- **bgp**—Border Gateway Protocol
- **dvmrp**—Distance Vector Multicast Routing Protocol
- **msdp**—Multicast Source Discovery Protocol
- **pim**—Protocol Independent Multicast
- **rip**—Routing Information Protocol
- **ripng**—Routing Information Protocol next generation

Additional Information Routes displayed are routes that the routing table has exported into the routing protocol and that have been filtered by the associated protocol's **export** routing policy statements.

Required Privilege Level view

Related Documentation

- *Example: Configuring the MED Attribute Directly*

List of Sample Output

show route advertising-protocol bgp (Layer 3 VPN) on page 510
show route advertising-protocol bgp detail on page 511
show route advertising-protocol bgp detail (Layer 2 VPN) on page 511
show route advertising-protocol bgp detail (Layer 3 VPN) on page 511
show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address) on page 511

Output Fields Table 30 on page 509 lists the output fields for the **show route advertising-protocol** command. Output fields are listed in the approximate order in which they appear.

Table 30: show route advertising-protocol Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, inet.0.	All levels
number **destinations**	Number of destinations for which there are routes in the routing table.	All levels
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)	All levels
Prefix	Destination prefix.	**brief** none
destination-prefix **(entry , announced)**	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
BGP group and type	BGP group name and type (**Internal** or **External**).	**detail extensive**
Route Distinguisher	Unique 64-bit prefix augmenting each IP subnet.	**detail extensive**
Advertised Label	Incoming label advertised by the LDP. When an IP packet enters a label-switched path (LSP), the ingress router examines the packet and assigns it a label based on its destination, placing the label in the packet's header. The label transforms the packet from one that is forwarded based on its IP routing information to one that is forwarded based on information associated with the label.	**detail extensive**
Label-Base, range	First label in a block of labels and label block size. A remote PE router uses this first label when sending traffic toward the advertising PE router.	**detail extensive**
VPN Label	Virtual private network (VPN) label. Packets are sent between CE and PE routers by advertising VPN labels. VPN labels transit over either an RSVP or an LDP LSP tunnel.	**detail extensive**
Nexthop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route. If the next-hop advertisement to the peer is **Self**, and the RIB-out next hop is a specific IP address, the RIB-out IP address is included in the extensive output. See show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address) on page 511.	All levels
MED	Multiple exit discriminator value included in the route.	**brief**
Lclpref or **Localpref**	Local preference value included in the route.	All levels

Table 30: show route advertising-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if configured on the router, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
Communities	Community path attribute for the route. See the output field table for the show route detail command for all possible values for this field.	**detail extensive**
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.	**detail extensive**
Attrset AS	Number, local preference, and path of the autonomous system (AS) that originated the route. These values are stored in the **Attrset** attribute at the originating router.	**detail extensive**
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).	**detail extensive**
control flags	Control flags: **none** or **Site Down**.	**detail extensive**
mtu	Maximum transmission unit (MTU) of the Layer 2 circuit.	**detail extensive**

Sample Output

show route advertising-protocol bgp (Layer 3 VPN)

```
user@host> show route advertising-protocol bgp 10.255.14.171
      VPN-A.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix             Nexthop                MED    Lclpref AS path
      10.255.14.172/32   Self                     1        100 I
      VPN-B.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix             Nexthop                MED    Lclpref AS path
      10.255.14.181/32   Self                     2        100 I
```

show route advertising-protocol bgp detail

```
user@host> show route advertising-protocol bgp 111.222.1.3 detail
bgp20.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
111.222.1.11/32 (1 entry, 1 announced)
 BGP group pe-pe type Internal
     Route Distinguisher: 111.255.14.11:69
     Advertised Label: 100000
     next hop: Self
     Localpref: 100
     AS path: 2 I
     Communities: target:69:20
     AIGP 210
111.8.0.0/16 (1 entry, 1 announced)
 BGP group pe-pe type Internal
     Route Distinguisher: 111.255.14.11:69
     Advertised Label: 100000
     Next hop: Self
     Localpref: 100
     AS path: 2 I
     Communities: target:69:20
     AIGP 210
```

show route advertising-protocol bgp detail (Layer 2 VPN)

```
user@host> show route advertising-protocol bgp 192.168.24.1 detail
vpn-a.l2vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
192.168.16.1:1:1:1/96 (1 entry, 1 announced)
 BGP group int type Internal
     Route Distinguisher: 192.168.16.1:1
     Label-base : 32768, range : 3
     Nexthop: Self
     Localpref: 100
     AS path: I
     Communities: target:65412:100
     AIGP 210
     Layer2-info: encaps:VLAN, control flags:, mtu:
```

show route advertising-protocol bgp detail (Layer 3 VPN)

```
user@host> show route advertising-protocol bgp 10.255.14.176 detail
vpna.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 10.49.0.0/30 (1 entry, 1 announced)
 BGP group ibgp type Internal
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101264
     Nexthop: Self
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AIGP 210
     AttrSet AS: 100
         Localpref: 100
         AS path: I
...
```

show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address)

```
user@host> show route advertising-protocol bgp 200.0.0.2 170.0.1.0/24 extensive all
inet.0: 13 destinations, 19 routes (13 active, 0 holddown, 6 hidden)
  170.0.1.0/24 (2 entries, 1 announced)
```

```
 BGP group eBGP-INTEROP type External
     Nexthop: Self (rib-out 10.100.3.2)
     AS path: [4713] 200 I
...
```

show route all

Syntax
show route all
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route all

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display information about all routes in all routing tables, including private, or internal, tables.

Options
none—Display information about all routes in all routing tables, including private, or internal, tables.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route all on page 513

Output Fields
In Junos OS Release 9.5 and later, only the output fields for the **show route all** command display all routing tables, including private, or hidden, routing tables. The output field table of the show route command does not display entries for private, or hidden, routing tables in Junos OS Release 9.5 and later.

Sample Output

show route all

The following example displays a snippet of output from the **show route** command and then displays the same snippet of output from the **show route all** command:

```
user@host> show route
mpls.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
0                  *[MPLS/0] 2d 02:24:39, metric 1
                      Receive
1                  *[MPLS/0] 2d 02:24:39, metric 1
                      Receive
2                  *[MPLS/0] 2d 02:24:39, metric 1
                      Receive
800017             *[VPLS/7] 1d 14:00:16
                    > via vt-3/2/0.32769, Pop
800018             *[VPLS/7] 1d 14:00:26
                    > via vt-3/2/0.32772, Pop

user@host> show route all
mpls.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
Restart Complete
```

```
+ = Active Route, - = Last Active, * = Both
0                  *[MPLS/0] 2d 02:19:12, metric 1
                      Receive
1                  *[MPLS/0] 2d 02:19:12, metric 1
                      Receive
2                  *[MPLS/0] 2d 02:19:12, metric 1
                      Receive
800017             *[VPLS/7] 1d 13:54:49
                    > via vt-3/2/0.32769, Pop
800018             *[VPLS/7] 1d 13:54:59
                    > via vt-3/2/0.32772, Pop
vt-3/2/0.32769      [VPLS/7] 1d 13:54:49
                      Unusable
vt-3/2/0.32772      [VPLS/7] 1d 13:54:59
                      Unusable
```

show route aspath-regex

Syntax

show route aspath-regex *regular-expression*
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route aspath-regex *regular-expression*

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the entries in the routing table that match the specified autonomous system (AS) path regular expression.

Options

regular-expression—Regular expression that matches an entire AS path.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information

You can specify a regular expression as:

- An individual AS number
- A period wildcard used in place of an AS number
- An AS path regular expression that is enclosed in parentheses

You also can include the operators described in the table of AS path regular expression operators in the *Junos Policy Framework Configuration Guide*. The following list summarizes these operators:

- ***{m,n}***—At least *m* and at most *n* repetitions of the AS path term.
- ***{m}***—Exactly *m* repetitions of the AS path term.
- ***{m,}***—*m* or more repetitions of the AS path term.
- *******—Zero or more repetitions of an AS path term.
- ***+***—One or more repetitions of an AS path term.
- ***?***—Zero or one repetition of an AS path term.
- ***aspath_term | aspath_term***—Match one of the two AS path terms.

  When you specify more than one AS number or path term, or when you include an operator in the regular expression, enclose the entire regular expression in quotation marks. For example, to match any path that contains AS number 234, specify the following command:

  ```
  show route aspath-regex ".* 234 .*"
  ```

Required Privilege Level

view

Related Documentation
- Example: Using AS Path Regular Expressions on page 283

List of Sample Output
show route aspath-regex (Matching a Specific AS Number) on page 516
show route aspath-regex (Matching Any Path with Two AS Numbers) on page 516

Output Fields
For information about output fields, see the output field table for the show route command.

Sample Output

show route aspath-regex (Matching a Specific AS Number)

```
user@host> show route aspath-regex 65477
inet.0: 46411 destinations, 46411 routes (46409 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

111.222.1.0/25     *[BGP/170] 00:08:48, localpref 100, from 111.222.2.24
                   AS Path: [65477] ({65488 65535}) IGP
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
111.222.1.128/25   *[IS-IS/15] 09:15:37, metric 37, tag 1
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
                    [BGP/170] 00:08:48, localpref 100, from 111.222.2.24
                   AS Path: [65477] ({65488 65535}) IGP
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
...
```

show route aspath-regex (Matching Any Path with Two AS Numbers)

```
user@host> show route aspath-regex ?.* 234 3561 .*?

inet.0: 46351 destinations, 46351 routes (46349 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

9.20.0.0/17        *[BGP/170] 01:35:00, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 2685 2686 Incomplete
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
12.10.231.0/24     *[BGP/170] 01:35:00, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 5696 7369 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
24.64.32.0/19      *[BGP/170] 01:34:59, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 6327 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
...
```

show route best

Syntax
show route best *destination-prefix*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route best *destination-prefix*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the route in the routing table that is the best route to the specified address or range of addresses. The best route is the longest matching route.

Options
brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

destination-prefix—Address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route best on page 517
show route best detail on page 518
show route best extensive on page 519
show route best terse on page 519

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route best

```
user@host> show route best 10.255.70.103
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.255.70.103/32   *[OSPF/10] 1d 13:19:20, metric 2
                    > to 10.31.1.6 via ge-3/1/0.0
                      via so-0/3/0.0

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.255.70.103/32   *[RSVP/7] 1d 13:20:13, metric 2
                    > via so-0/3/0.0, label-switched-path green-r1-r3

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
10.0.0.0/8          *[Direct/0] 2d 01:43:34
                     > via fxp2.0
                     [Direct/0] 2d 01:43:34
                     > via fxp1.0
```

show route best detail

```
user@host> show route best 10.255.70.103 detail
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
10.255.70.103/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: 10.31.1.6 via ge-3/1/0.0, selected
                Next hop: via so-0/3/0.0
                State: <Active Int>
                Local AS:    69
                Age: 1d 13:20:06        Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
10.255.70.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 5
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100016
                State: <Active Int>
                Local AS:    69
                Age: 1d 13:20:59        Metric: 2
                Task: RSVP
                Announcement bits (1): 1-Resolve tree 2
                AS path: I

private1__inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
10.0.0.0/8 (2 entries, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp2.0, selected
                State: <Active Int>
                Age: 2d 1:44:20
                Task: IF
                AS path: I
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp1.0, selected
                State: <NotBest Int>
                Inactive reason: No difference
                Age: 2d 1:44:20
                Task: IF
                AS path: I
```

show route best extensive

The output for the **show route best extensive** command is identical to that for the **show route best detail** command. For sample output, see show route best detail on page 518.

show route best terse

```
user@host> show route best 10.255.70.103 terse
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.103/32   O  10          2            >10.31.1.6
                                                   so-0/3/0.0

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.103/32   R   7          2            >so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.0.0.0/8         D   0                     >fxp2.0
                     D   0                     >fxp1.0
```

show route brief

Syntax
show route brief
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route brief
<*destination-prefix*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display brief information about the active entries in the routing tables.

Options
none—Display all active entries in the routing table.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route brief on page 520

Output Fields
For information about output fields, see the Output Field table of the show route command.

Sample Output

show route brief

```
user@host> show route brief
inet.0: 10 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 1w5d 20:30:29
                      Discard
10.255.245.51/32   *[Direct/0] 2w4d 13:11:14
                    > via lo0.0
172.16.0.0/12      *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.0.0/18     *[Static/5] 1w5d 20:30:29
                    > to 192.168.167.254 via fxp0.0
192.168.40.0/22    *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.64.0/18    *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.164.0/22   *[Direct/0] 2w4d 13:11:14
                    > via fxp0.0
192.168.164.51/32  *[Local/0] 2w4d 13:11:14
                      Local via fxp0.0
207.17.136.192/32  *[Static/5] 2w4d 13:11:14
```

```
                    > to 192.168.167.254 via fxp0.0
green.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
100.101.0.0/16     *[Direct/0] 1w5d 20:30:28
                    > via fe-0/0/3.0
100.101.2.3/32     *[Local/0] 1w5d 20:30:28
                      Local via fe-0/0/3.0
224.0.0.5/32       *[OSPF/10] 1w5d 20:30:29, metric 1
                      MultiRecv
```

show route community

Syntax
show route community *as-number:community-value*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route community *as-number:community-value*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the route entries in each routing table that are members of a Border Gateway Protocol (BGP) community.

Options
as-number:community-value—One or more community identifiers. ***as-number*** is the AS number, and ***community-value*** is the community identifier. When you specify more than one community identifier, enclose the identifiers in double quotation marks. Community identifiers can include wildcards.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information
Specifying the community option displays all routes matching the community found within the routing table. The community option does not limit the output to only the routes being advertised to the neighbor after any egress routing policy.

Required Privilege Level
view

Related Documentation
- show route detail on page 531

List of Sample Output
show route community on page 522

Output Fields
For information about output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route community

```
user@host> show route community 234:80
inet.0: 46511 destinations, 46511 routes (46509 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

4.0.0.0/8          *[BGP/170] 03:33:07, localpref 100, from 131.103.20.49
                   AS Path: {666} 234 2548 1 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
6.0.0.0/8          *[BGP/170] 03:33:07, localpref 100, from 131.103.20.49
```

```
                    AS Path: {666} 234 2548 568 721 Incomplete
                      to 192.156.169.1 via 192.156.169.14(so-0/0/0)
9.2.0.0/16          *[BGP/170] 03:33:06, localpref 100, from 131.103.20.49
                    AS Path: {666} 234 2548 1673 1675 1747 IGP
                      to 192.156.169.1 via 192.156.169.14(so-0/0/0)
```

show route community-name

Syntax show route community-name *community-name*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches) show route community-name *community-name*
<brief | detail | extensive | terse>

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description Display the route entries in each routing table that are members of a Border Gateway Protocol (BGP) community, specified by a community name.

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

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level view

List of Sample Output show route community-name on page 524

Output Fields For information about output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route community-name

```
user@host> show route community-name red-com
inet.0: 17 destinations, 17 routes (16 active, 0 holddown, 1 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

instance1.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.245.212/32  *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: 300 I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix
20.20.20.20/32     *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix
100.1.4.0/24       *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix
```

```
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.245.204:10:10.255.245.212/32
                   *[BGP/170] 00:06:40, localpref 100, from 10.255.245.204
                      AS path: 300 I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix
10.255.245.204:10:20.20.20.20/32
                   *[BGP/170] 00:36:02, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix
10.255.245.204:10:100.1.4.0/24
                   *[BGP/170] 00:36:02, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 100.1.2.2 via ge-1/1/0.0, label-switched-path to_fix

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

instance1.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route damping

Syntax
show route damping (decayed | history | suppressed)
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switch and QFX Series)
show route damping (decayed | history | suppressed)
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.

Description
Display the BGP routes for which updates might have been reduced because of route flap damping.

Options
brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

decayed—Display route damping entries that might no longer be valid, but are not suppressed.

history—Display entries that have already been withdrawn, but have been logged.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

suppressed—Display entries that have been suppressed and are no longer being installed into the forwarding table or exported by routing protocols.

Required Privilege Level
view

Related Documentation
- *clear bgp damping*
- show policy damping on page 496

List of Sample Output
show route damping decayed detail on page 529
show route damping history on page 530
show route damping history detail on page 530

Output Fields
Table 31 on page 526 lists the output fields for the **show route damping** command. Output fields are listed in the approximate order in which they appear.

Table 31: show route damping Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, **inet.0**.	All levels
destinations	Number of destinations for which there are routes in the routing table.	All levels

Table 31: show route damping Output Fields *(continued)*

Field Name	Field Description	Level of Output
***number* routes**	Number of routes in the routing table and total number of routes in the following states: • **active** • **holddown** (routes that are in a pending state before being declared inactive) • **hidden** (the routes are not used because of a routing policy)	All levels
***destination-prefix* (entry, announced)**	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • ***—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the LocalPref value for Route 1 is 100, the Preference2 value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.	All levels
Next-hop reference count	Number of references made to the next hop.	**detail extensive**
Source	IP address of the route source.	**detail extensive**
Next hop	Network layer address of the directly reachable neighboring system.	**detail extensive**
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**.	**detail extensive**
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.	**detail extensive**
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.	**detail extensive**
State	Flags for this route. For a description of possible values for this field, see the output field table for the show route detail command.	**detail extensive**
Local AS	AS number of the local routing device.	**detail extensive**
Peer AS	AS number of the peer routing device.	**detail extensive**

Table 31: show route damping Output Fields *(continued)*

Field Name	Field Description	Level of Output
Age	How long the route has been known.	**detail extensive**
Metric	Metric for the route.	**detail extensive**
Task	Name of the protocol that has added the route.	**detail extensive**
Announcement bits	List of protocols that announce this route. ***n*-Resolve inet** indicates that the route is used for route resolution for next hops found in the routing table. ***n*** is an index used by Juniper Networks customer support only.	**detail extensive**
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
to	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.	**brief** none
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**.	**brief** none
Communities	Community path attribute for the route. See the output field table for the show route detail command.	**detail extensive**
Localpref	Local preference value included in the route.	All levels
Router ID	BGP router ID as advertised by the neighbor in the open message.	**detail extensive**
Merit (last update/now)	Last updated and current figure-of-merit value.	**detail extensive**

Table 31: show route damping Output Fields *(continued)*

Field Name	Field Description	Level of Output
damping-parameters	Name that identifies the damping parameters used, which is defined in the damping statement at the **[edit policy-options]** hierarchy level.	**detail extensive**
Last update	Time of most recent change in path attributes.	**detail extensive**
First update	Time of first change in path attributes, which started the route damping process.	**detail extensive**
Flaps	Number of times the route has gone up or down or its path attributes have changed.	**detail extensive**
Suppressed	(**suppressed** keyword only) This route is currently suppressed. A suppressed route does not appear in the forwarding table and routing protocols do not export it.	All levels
Reusable in	(**suppressed** keyword only) Time when a suppressed route will again be available.	All levels
Preference will be	(**suppressed** keyword only) Preference value that will be applied to the route when it is again active.	All levels

Sample Output

show route damping decayed detail

```
user@host> show route damping decayed detail
inet.0: 173319 destinations, 1533668 routes (172625 active, 4 holddown, 108083
hidden)
10.0.111.0/24 (7 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 151973
                Source: 172.23.2.129
                Next hop: via so-1/2/0.0
                Next hop: via so-5/1/0.0, selected
                Next hop: via so-6/0/0.0
                Protocol next hop: 172.23.2.129
                Indirect next hop: 89a1a00 264185
                State: <Active Ext>
                Local AS: 65000 Peer AS:   65490
                Age: 3:28       Metric2: 0
                Task: BGP_65490.172.23.2.129+179
                Announcement bits (6): 0-KRT 1-RT 4-KRT 5-BGP.0.0.0.0+179

             6-Resolve tree 2 7-Resolve tree 3
                AS path: 65490 65520 65525 65525 65525 65525 I ()
                Communities: 65501:390 65501:2000 65501:3000 65504:701
                Localpref: 100
                Router ID: 172.23.2.129
                Merit (last update/now): 1934/1790
                damping-parameters: damping-high
                Last update:       00:03:28 First update:       00:06:40
                Flaps: 2
```

show route damping history

```
user@host> show route damping history
inet.0: 173320 destinations, 1533529 routes (172624 active, 6 holddown, 108122
hidden)
+ = Active Route, - = Last Active, * = Both

10.108.0.0/15       [BGP ] 2d 22:47:58, localpref 100
                      AS path: 65220 65501 65502 I
                    > to 192.168.60.85 via so-3/1/0.0
```

show route damping history detail

```
user@host> show route damping history detail
inet.0: 173319 destinations, 1533435 routes (172627 active, 2 holddown, 108105
hidden)
10.108.0.0/15 (3 entries, 1 announced)
         BGP                 /-101
                Next-hop reference count: 69058
                Source: 192.168.60.85
                Next hop: 192.168.60.85 via so-3/1/0.0, selected
                State: <Hidden Ext>
                Inactive reason: Unusable path
                Local AS: 65000 Peer AS:  65220
                Age: 2d 22:48:10
                Task: BGP_65220.192.168.60.85+179
                AS path: 65220 65501 65502 I ()
                Communities: 65501:390 65501:2000 65501:3000 65504:3561
                Localpref: 100
                Router ID: 192.168.80.25
                Merit (last update/now): 1000/932
                damping-parameters: set-normal
                Last update:       00:01:05 First update:       00:01:05
                Flaps: 1
```

show route detail

Syntax
show route detail
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route detail
<*destination-prefix*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display detailed information about the active entries in the routing tables.

Options
none—Display all active entries in the routing table on all systems.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route detail on page 539
show route detail (with BGP Multipath) on page 545
show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs) on page 545

Output Fields
Table 32 on page 531 describes the output fields for the **show route detail** command. Output fields are listed in the approximate order in which they appear.

Table 32: show route detail Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)

Table 32: show route detail Output Fields *(continued)*

Field Name	Field Description

route-destination **(entry, announced)**	Route destination (for example:10.0.0.1/24). The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:
	• ***MPLS-label*** (for example, 80001).
	• ***interface-name*** (for example, ge-1/0/2).
	• ***neighbor-address*****:*****control-word-status*****:*****encapsulation type*****:*****vc-id*****:*****source*** (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).
	  ◦ ***neighbor-address***—Address of the neighbor.
	  ◦ ***control-word-status***—Whether the use of the control word has been negotiated for this virtual circuit: **NoCtrlWord** or **CtrlWord**.
	  ◦ ***encapsulation type***—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
	  ◦ ***vc-id***—Virtual circuit identifier.
	  ◦ ***source***—Source of the advertisement: **Local** or **Remote**.

label stacking	**(**Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels. • **S=0 route** indicates that a packet with an incoming label stack depth of 2 or more exits this routing device with one fewer label (the label-popping operation is performed). • If there is no **S=** information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).
[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.
Level	(IS-IS only). In IS-IS, a single AS can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
Next-hop type	Type of next hop. For a description of possible values for this field, see Table 33 on page 535.

Table 32: show route detail Output Fields *(continued)*

Field Name	Field Description
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path *lsp-path-name*	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.
State	State of the route (a route can be in more than one state). See Table 34 on page 537.
Local AS	AS number of the local routing device.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric*n*	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.

Table 32: show route detail Output Fields *(continued)*

Field Name	Field Description
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances. For sample output, see show route table.
Task	Name of the protocol that has added the route.
Announcement bits	List of protocols that announce this route. ***n*-Resolve inet** indicates that the route is used for route resolution for next hops found in the routing table. ***n*** is an index used by Juniper Networks customer support only.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used in the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the routing device, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
FECs bound to route	Point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See Table 35 on page 539 for all possible values for this field.

Table 32: show route detail Output Fields *(continued)*

Field Name	Field Description
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: **none** or **Site Down**.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Accepted Multipath	Current active path when BGP multipath is configured.
Accepted MultipathContrib	Path currently contributing to BGP multipath.
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.

Table 33 on page 535 describes all possible values for the Next-hop Types output field.

Table 33: Next-hop Types Output Field Values

Next-Hop Type	Description
Broadcast (bcast)	Broadcast next hop.
Deny	Deny next hop.
Discard	Discard next hop.
Flood	Flood next hop. Consists of components called branches, up to a maximum of 32 branches. Each flood next-hop branch sends a copy of the traffic to the forwarding interface. Used by point-to-multipoint RSVP, point-to-multipoint LDP, point-to-multipoint CCC, and multicast.
Hold	Next hop is waiting to be resolved into a unicast or multicast type.
Indexed (idxd)	Indexed next hop.

Table 33: Next-hop Types Output Field Values *(continued)*

Next-Hop Type	Description
Indirect (indr)	Used with applications that have a protocol next hop address that is remote. You are likely to see this next-hop type for internal BGP (IBGP) routes when the BGP next hop is a BGP neighbor that is not directly connected.
Interface	Used for a network address assigned to an interface. Unlike the router next hop, the interface next hop does not reference any specific node on the network.
Local (locl)	Local address on an interface. This next-hop type causes packets with this destination address to be received locally.
Multicast (mcst)	Wire multicast next hop (limited to the LAN).
Multicast discard (mdsc)	Multicast discard.
Multicast group (mgrp)	Multicast group member.
Receive (recv)	Receive.
Reject (rjct)	Discard. An ICMP unreachable message was sent.
Resolve (rslv)	Resolving next hop.
Routed multicast (mcrt)	Regular multicast next hop.
Router	A specific node or set of nodes to which the routing device forwards packets that match the route prefix. To qualify as next-hop type router, the route must meet the following criteria: • Must not be a direct or local subnet for the routing device. • Must have a next hop that is directly connected to the routing device.
Table	Routing table next hop.
Unicast (ucst)	Unicast.
Unilist (ulst)	List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.

Table 34 on page 537 describes all possible values for the State output field. A route can be in more than one state (for example, **<Active NoReadvrt Int Ext>**).

Table 34: State Output Field Values

Value	Description
Accounting	Route needs accounting.
Active	Route is active.
Always Compare MED	Path with a lower multiple exit discriminator (MED) is available.
AS path	Shorter AS path is available.
Cisco Non-deterministic MED selection	Cisco nondeterministic MED is enabled, and a path with a lower MED is available.
Clone	Route is a clone.
Cluster list length	Length of cluster list sent by the route reflector.
Delete	Route has been deleted.
Ex	Exterior route.
Ext	BGP route received from an external BGP neighbor.
FlashAll	Forces all protocols to be notified of a change to any route, active or inactive, for a prefix. When not set, protocols are informed of a prefix only when the active route changes.
Hidden	Route not used because of routing policy.
IfCheck	Route needs forwarding RPF check.
IGP metric	Path through next hop with lower IGP metric is available.
Inactive reason	Flags for this route, which was not selected as best for a particular destination.
Initial	Route being added.
Int	Interior route.
Int Ext	BGP route received from an internal BGP peer or a BGP confederation peer.
Interior > Exterior > Exterior via Interior	Direct, static, IGP, or EBGP path is available.
Local Preference	Path with a higher local preference value is available.
Martian	Route is a martian (ignored because it is obviously invalid).

Table 34: State Output Field Values *(continued)*

Value	Description
MartianOK	Route exempt from martian filtering.
Next hop address	Path with lower metric next hop is available.
No difference	Path from neighbor with lower IP address is available.
NoReadvrt	Route not to be advertised.
NotBest	Route not chosen because it does not have the lowest MED.
Not Best in its group	Incoming BGP AS is not the best of a group (only one AS can be the best).
NotInstall	Route not to be installed in the forwarding table.
Number of gateways	Path with a greater number of next hops is available.
Origin	Path with a lower origin code is available.
Pending	Route pending because of a hold-down configured on another route.
Release	Route scheduled for release.
RIB preference	Route from a higher-numbered routing table is available.
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.
Route Metric or MED comparison	Route with a lower metric or MED is available.
Route Preference	Route with lower preference value is available
Router ID	Path through a neighbor with lower ID is available.
Secondary	Route not a primary route.
Unusable path	Path is not usable because of one of the following conditions: • The route is damped. • The route is rejected by an import policy. • The route is unresolved.
Update source	Last tiebreaker is the lowest IP address value.

Table 35 on page 539 describes the possible values for the Communities output field.

Table 35: Communities Output Field Values

Value	Description
area-number	4 bytes, encoding a 32-bit area number. For AS-external routes, the value is **0**. A nonzero value identifies the route as internal to the OSPF domain, and as within the identified area. Area numbers are relative to a particular OSPF domain.
bandwidth: ***local AS number:link-bandwidth-number***	Link-bandwidth community value used for unequal-cost load balancing. When BGP has several candidate paths available for multipath purposes, it does not perform unequal-cost load balancing according to the link-bandwidth community unless all candidate paths have this attribute.
domain-id	Unique configurable number that identifies the OSPF domain.
domain-id-vendor	Unique configurable number that further identifies the OSPF domain.
link-bandwidth-number	Link-bandwidth number: from **0** through **4,294,967,295** (bytes per second).
local AS number	Local AS number: from **1** through **65,535**.
options	1 byte. Currently this is only used if the route type is **5** or **7**. Setting the least significant bit in the field indicates that the route carries a type 2 metric.
origin	(Used with VPNs) Identifies where the route came from.
ospf-route-type	1 byte, encoded as **1** or **2** for intra-area routes (depending on whether the route came from a type 1 or a type 2 LSA); **3** for summary routes; **5** for external routes (area number must be**0**); **7** for NSSA routes; or **129** for sham link endpoint addresses.
route-type-vendor	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x8000**. The format is ***area-number:ospf-route-type:options***.
rte-type	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x0306**. The format is ***area-number:ospf-route-type:options***.
target	Defines which VPN the route participates in; **target** has the format ***32-bit IP address:16-bit number***. For example, 10.19.0.0:100.
unknown IANA	Incoming IANA codes with a value between **0x1** and **0x7fff**. This code of the BGP extended community attribute is accepted, but it is not recognized.
unknown OSPF vendor community	Incoming IANA codes with a value above **0x8000**. This code of the BGP extended community attribute is accepted, but it is not recognized.

Sample Output

show route detail

```
user@host> show route detail

inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
```

```
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 1:31:43
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

10.31.1.0/30 (2 entries, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 2
                Next hop: via so-0/3/0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:30:17
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Local AS:    69
                Age: 1:30:17    Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

10.31.1.1/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 7
                Interface: so-0/3/0.0
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:30:20
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I

...

10.31.2.0/30 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: via so-0/3/0.0
                Next hop: 10.31.1.6 via ge-3/1/0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:29:56    Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...
```

```
224.0.0.2/32 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

224.0.0.22/32 (1 entry, 1 announced)
        *IGMP   Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:43
                Task: IGMP
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

10.255.70.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100096
                State: <Active Int>
                Local AS:    69
                Age: 1:25:49    Metric: 2
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

10.255.71.238/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r2
                State: <Active Int>
                Local AS:    69
                Age: 1:25:49    Metric: 1
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

private__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

47.0005.80ff.f800.0000.0108.0001.0102.5507.1052/152 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
```

```
                State: <Active Int>
                Local AS:    69
                Age: 1:31:44
                Task: IF
                AS path: I

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
0   (1 entry, 1 announced)
        *MPLS   Preference: 0
                Next hop type: Receive
                Next-hop reference count: 6
                State: <Active Int>
                Local AS:    69
                Age: 1:31:45    Metric: 1
                Task: MPLS
                Announcement bits (1): 0-KRT
                AS path: I

...

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
299776 (1 entry, 1 announced)
TSI:
KRT in-kernel 299776 /52 -> {Flood}
        *RSVP   Preference: 7
                Next hop type: Flood
                Next-hop reference count: 130
                Flood nexthop branches exceed maximum
                Address: 0x8ea65d0
...

800010 (1 entry, 1 announced)
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: via vt-3/2/0.32769, selected
                Label operation: Pop
                State: <Active Int>
                Age: 1:29:30
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

vt-3/2/0.32769 (1 entry, 1 announced)
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 800012, Push 100096(top)
                Protocol next hop: 10.255.70.103
                Push 800012
                Indirect next hop: 87272e4 1048574
                State: <Active Int>
                Age: 1:29:30    Metric2: 2
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 1-Common L2 VC
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
```

```
abcd::10:255:71:52/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:31:44
                Task: IF
                AS path: I

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:44
                Task: IF
                AS path: I

ff02::2/128 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::d/128 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::16/128 (1 entry, 1 announced)
        *MLD    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:43
                Task: MLD
                Announcement bits (1): 0-KRT
                AS path: I

private.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.16385, selected
                State: <Active NoReadvrt Int>
                Age: 1:31:44
```

```
                Task: IF
                AS path: I

green.l2vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

10.255.70.103:1:3:1/96 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.70.103:1
                Next-hop reference count: 7
                Source: 10.255.70.103
                Protocol next hop: 10.255.70.103
                Indirect next hop: 2 no-forward
                State: <Secondary Active Int Ext>
                Local AS:    69 Peer AS:    69
                Age: 1:25:49    Metric2: 1
                AIGP 210
                Task: BGP_69.10.255.70.103+179
                Announcement bits (1): 0-green-l2vpn
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Label-base: 800008, range: 8
                Localpref: 100
                Router ID: 10.255.70.103
                Primary Routing Table bgp.l2vpn.0

10.255.71.52:1:1:1/96 (1 entry, 1 announced)
        *L2VPN  Preference: 170/-1
                Next-hop reference count: 5
                Protocol next hop: 10.255.71.52
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:31:40    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:Site-Down,
                mtu: 0
                Label-base: 800016, range: 8, status-vector: 0x9F

10.255.71.52:1:5:1/96 (1 entry, 1 announced)
        *L2VPN  Preference: 170/-101
                Next-hop reference count: 5
                Protocol next hop: 10.255.71.52
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:31:40    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:, mtu: 0
                Label-base: 800008, range: 8, status-vector: 0x9F

...

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.245.255.63:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
```

```
                        Protocol next hop: 10.245.255.63 Indirect next hop: 86af000 296
                        State: <Active Int>
                        Local AS:    99
                        Age: 10:21
                        Task: l2 circuit
                        Announcement bits (1): 0-LDP
                        AS path: I
                        VC Label 100000, MTU 1500, VLAN ID 512
```

show route detail (with BGP Multipath)

```
user@host> show route detail

10.1.1.8/30 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 262142
                Address: 0x901a010
                Next-hop reference count: 2
                Source: 10.1.1.2
                Next hop: 10.1.1.2 via ge-0/3/0.1, selected
                Next hop: 10.1.1.6 via ge-0/3/0.5
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 5:04:43
                Task: BGP_2.10.1.1.2+59955
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted Multipath
                Localpref: 100
                Router ID: 1.1.1.2
         BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 678
                Address: 0x8f97520
                Next-hop reference count: 9
                Source: 10.1.1.6
                Next hop: 10.1.1.6 via ge-0/3/0.5, selected
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Active preferred
                Local AS:     1 Peer AS:     2
                Age: 5:04:43
                Task: BGP_2.10.1.1.6+58198
                AS path: 2 I
                Accepted MultipathContrib
                Localpref: 100
                Router ID: 1.1.1.3
```

show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs)

```
user@host> show route label 299872 detail
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
299872 (1 entry, 1 announced)
        *LDP    Preference: 9
                Next hop type: Flood
                Next-hop reference count: 3
                Address: 0x9097d90
                Next hop: via vt-0/1/0.1
                Next-hop index: 661
                Label operation: Pop
                Address: 0x9172130
                Next hop: via so-0/0/3.0
                Next-hop index: 654
```

```
                    Label operation: Swap 299872
                    State: **Active Int>
                    Local AS:  1001
                    Age: 8:20       Metric: 1
                    Task: LDP
                    Announcement bits (1): 0-KRT
                    AS path: I
                   FECs bound to route: P2MP root-addr 10.255.72.166, grp 232.1.1.1,
src 192.168.142.2
```

show route exact

Syntax
show route exact *destination-prefix*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route exact *destination-prefix*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display only the routes that exactly match the specified address or range of addresses.

Options
brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

destination-prefix—Address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route exact on page 547
show route exact detail on page 547
show route exact extensive on page 548
show route exact terse on page 548

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route exact

```
user@host> show route exact 207.17.136.0/24

inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
207.17.136.0/24    *[Static/5] 2d 03:30:22
                    > to 192.168.71.254 via fxp0.0
```

show route exact detail

```
user@host> show route exact 207.17.136.0/24 detail

inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
207.17.136.0/24 (1 entry, 1 announced)
        *Static Preference: 5
```

```
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2d 3:30:26
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I
```

show route exact extensive

```
user@host> show route exact 207.17.136.0/24 extensive
inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
207.17.136.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.0/24 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 1:25:18
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I
```

show route exact terse

```
user@host> show route exact 207.17.136.0/24 terse

inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 207.17.136.0/24    S   5                      >192.168.71.254
```

show route export

Syntax
show route export
<brief | detail>
<instance <*instance-name*> | *routing-table-name*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route export
<brief | detail>
<instance <*instance-name*> | *routing-table-name*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display policy-based route export information. Policy-based export simplifies the process of exchanging route information between routing instances.

Options
none—(Same as **brief**.) Display standard information about policy-based export for all instances and routing tables on all systems.

brief | detail—(Optional) Display the specified level of output.

instance <*instance-name*>—(Optional) Display a particular routing instance for which policy-based export is currently enabled.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

routing-table-name—(Optional) Display information about policy-based export for all routing tables whose name begins with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route export inet** command).

Required Privilege Level
view

List of Sample Output
show route export on page 550
show route export detail on page 550
show route export instance detail on page 550

Output Fields
Table 36 on page 549 lists the output fields for the **show route export** command. Output fields are listed in the approximate order in which they appear.

Table 36: show route export Output Fields

Field Name	Field Description	Level of Output
Table or *table-name*	Name of the routing tables that either import or export routes.	All levels
Routes	Number of routes exported from this table into other tables. If a particular route is exported to different tables, the counter will only increment by one.	**brief** none
Export	Whether the table is currently exporting routes to other tables: **Y** or **N** (Yes or No).	**brief** none

Table 36: show route export Output Fields *(continued)*

Field Name	Field Description	Level of Output
Import	Tables currently importing routes from the originator table. (Not displayed for tables that are not exporting any routes.)	**detail**
Flags	(**instance** keyword only) Flags for this feature on this instance: • **config auto-policy**—The policy was deduced from the configured IGP export policies. • **cleanup**—Configuration information for this instance is no longer valid. • **config**—The instance was explicitly configured.	**detail**
Options	(**instance** keyword only) Configured option displays the type of routing tables the feature handles: • **unicast**—Indicates *instance*.inet.0. • **multicast**—Indicates *instance*.inet.2. • **unicast multicast**—Indicates *instance*.inet.0 and *instance*.inet.2.	**detail**
Import policy	(**instance** keyword only) Policy that **route export** uses to construct the import-export matrix. Not displayed if the instance type is **vrf**.	**detail**
Instance	(**instance** keyword only) Name of the routing instance.	**detail**
Type	(**instance** keyword only) Type of routing instance: **forwarding**, **non-forwarding**, or **vrf**.	**detail**

Sample Output

show route export

```
user@host> show route export
Table                            Export           Routes
inet.0                           N                     0
black.inet.0                     Y                     3
red.inet.0                       Y                     4
```

show route export detail

```
user@host>  show route export detail
inet.0                                          Routes:        0
black.inet.0                                    Routes:        3
  Import: [ inet.0 ]
red.inet.0                                      Routes:        4
  Import: [ inet.0 ]
```

show route export instance detail

```
user@host>  show route export instance detail
Instance: master                          Type: forwarding
  Flags: <config auto-policy> Options: <unicast multicast>
  Import policy: [ (ospf-master-from-red || isis-master-from-black) ]
```

```
Instance: black                      Type: non-forwarding
Instance: red                        Type: non-forwarding
```

show route extensive

Syntax
show route extensive
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route extensive
<*destination-prefix*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display extensive information about the active entries in the routing tables.

Options
none—Display all active entries in the routing table.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route extensive on page 558
show route extensive (Access Route) on page 564
show route extensive (BGP PIC Edge) on page 565
show route extensive (FRR and LFA) on page 565
show route extensive (Route Reflector) on page 566
show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs) on page 566

Output Fields
Table 37 on page 552 describes the output fields for the **show route extensive** command. Output fields are listed in the approximate order in which they appear.

Table 37: show route extensive Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active). • **holddown** (routes that are in the pending state before being declared inactive). • **hidden** (routes that are not used because of a routing policy).

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description

route-destination **(entry, announced)**	Route destination (for example: 10.0.0.1/24). The **entry** value is the number of route for this destination, and the **announced** value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:
	• ***MPLS-label*** (for example, 80001).
	• ***interface-name*** (for example, ge-1/0/2).
	• ***neighbor-address*:*control-word-status*:*encapsulation type*:*vc-id*:*source*** (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).
	  ◦ ***neighbor-address***—Address of the neighbor.
	  ◦ ***control-word-status***—Whether the use of the control word has been negotiated for this virtual circuit: **NoCtrlWord** or **CtrlWord**.
	  ◦ ***encapsulation type***—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
	  ◦ ***vc-id***—Virtual circuit identifier.
	  ◦ ***source***—Source of the advertisement: **Local** or **Remote**.

TSI	Protocol header information.
label stacking	(Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels. • **S=0 route** indicates that a packet with an incoming label stack depth of two or more exits this router with one fewer label (the label-popping operation is performed). • If there is no **S=** information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).
[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.
Level	(IS-IS only). In IS-IS, a single autonomous system (AS) can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description
Next-hop type	Type of next hop. For a description of possible values for this field, see the Output Field table in the show route detail command.
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path *lsp-path-name*	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Offset	Whether the metric has been increased or decreased by an offset value.
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to recursively derive a forwarding next hop.
label-operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description
Indirect next hops	When present, a list of nodes that are used to resolve the path to the next-hop destination, in the order that they are resolved. When BGP PIC Edge is enabled, 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.
State	State of the route (a route can be in more than one state). See the Output Field table in the show route detail command.
Session ID	The BFD session ID number that represents the protection using MPLS fast reroute (FRR) and loop-free alternate (LFA).
Weight	Weight for the backup path. If the weight of an indirect next hop is larger than zero, the weight value is shown. For sample output, see show route table.

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description
Inactive reason	If the route is inactive, the reason for its current state is indicated. Typical reasons include: • **Active preferred**—Currently active route was selected over this route. • **Always compare MED**—Path with a lower multiple exit discriminator (MED) is available. • **AS path**—Shorter AS path is available. • **Cisco Non-deterministic MED selection**—Cisco nondeterministic MED is enabled and a path with a lower MED is available. • **Cluster list length**—Path with a shorter cluster list length is available. • **Forwarding use only**—Path is only available for forwarding purposes. • **IGP metric**—Path through the next hop with a lower IGP metric is available. • **IGP metric type**—Path with a lower OSPF link-state advertisement type is available. • **Interior > Exterior > Exterior via Interior**—Direct, static, IGP, or EBGP path is available. • **Local preference**—Path with a higher local preference value is available. • **Next hop address**—Path with a lower metric next hop is available. • **No difference**—Path from a neighbor with a lower IP address is available. • **Not Best in its group**—Occurs when multiple peers of the same external AS advertise the same prefix and are grouped together in the selection process. When this reason is displayed, an additional reason is provided (typically one of the other reasons listed). • **Number of gateways**—Path with a higher number of next hops is available. • **Origin**—Path with a lower origin code is available. • **OSPF version**—Path does not support the indicated OSPF version. • **RIB preference**—Route from a higher-numbered routing table is available. • **Route destinguisher**—64-bit prefix added to IP subnets to make them unique. • **Route metric or MED comparison**—Route with a lower metric or MED is available. • **Route preference**—Route with a lower preference value is available. • **Router ID**—Path through a neighbor with a lower ID is available. • **Unusable path**—Path is not usable because of one of the following conditions: the route is damped, the route is rejected by an import policy, or the route is unresolved. • **Update source**—Last tiebreaker is the lowest IP address value.
Local AS	Autonomous system (AS) number of the local routing device.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances. For sample output, see show route table.

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description
Task	Name of the protocol that has added the route.
Announcement bits	List of protocols that announce this route. ***n*-Resolve inet** indicates that the route is used for route resolution for next hops found in the routing table. ***n*** is an index used by Juniper Networks customer support only.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
FECs bound to route	Point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
AS path: I <Originator>	(For route reflected output only) Originator ID attribute set by the route reflector.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Cluster list	(For route reflected output only) Cluster ID sent by the route reflector.
Originator ID	(For route reflected output only) Address of router that originally sent the route to the route reflector.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See the Output Field table in the show route detail command for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).

Table 37: show route extensive Output Fields *(continued)*

Field Name	Field Description
control flags	Control flags: **none** or Site Down.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.
Originating RIB	Name of the routing table whose active route was used to determine the forwarding next-hop entry in the resolution database. For example, in the case of inet.0 resolving through inet.0 and inet.3, this field indicates which routing table, inet.0 or inet.3, provided the best path for a particular prefix.
Node path count	Number of nodes in the path.
Forwarding nexthops	Number of forwarding next hops. The forwarding next hop is the network layer address of the directly reachable neighboring system (if applicable) and the interface used to reach it.

Sample Output

show route extensive

```
user@host> show route extensive
inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.10.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 1:34:06
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

10.31.1.0/30 (2 entries, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 2
                Next hop: via so-0/3/0.0, selected
                State: <Active Int>
```

```
                Local AS:    69
                Age: 1:32:40
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Local AS:    69
                Age: 1:32:40    Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

10.31.1.1/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 7
                Interface: so-0/3/0.0
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:32:43
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I

...

10.31.2.0/30 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.31.2.0/30 -> {10.31.1.6}
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: via so-0/3/0.0
                Next hop: 10.31.1.6 via ge-3/1/0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:32:19    Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

224.0.0.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 224.0.0.2/32 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:34:08
                Task: PIM Recv
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...
```

```
224.0.0.22/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 224.0.0.22/32 -> {}
        *IGMP   Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:34:06
                Task: IGMP
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

10.255.70.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100096
                State: <Active Int>
                Local AS:    69
                Age: 1:28:12    Metric: 2
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

10.255.71.238/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r2
                State: <Active Int>
                Local AS:    69
                Age: 1:28:12    Metric: 1
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

...

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

47.0005.80ff.f800.0000.0108.0001.0102.5507.1052/152 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:34:07
                Task: IF
                AS path: I

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

0   (1 entry, 1 announced)
```

```
TSI:
KRT in-kernel 0      /36 -> {}
        *MPLS   Preference: 0
                Next hop type: Receive
                Next-hop reference count: 6
                State: <Active Int>
                Local AS:    69
                Age: 1:34:08    Metric: 1
                Task: MPLS
                Announcement bits (1): 0-KRT
                AS path: I

...

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
299776 (1 entry, 1 announced)
TSI:
KRT in-kernel 299776 /52 -> {Flood}
        *RSVP   Preference: 7
                Next hop type: Flood
                Next-hop reference count: 130
                Flood nexthop branches exceed maximum
                Address: 0x8ea65d0
...

800010 (1 entry, 1 announced)

TSI:
KRT in-kernel 800010 /36 -> {vt-3/2/0.32769}
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: via vt-3/2/0.32769, selected
                Label operation: Pop
                State: <Active Int>
                Age: 1:31:53
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

vt-3/2/0.32769 (1 entry, 1 announced)
TSI:
KRT in-kernel vt-3/2/0.32769.0      /16 -> {indirect(1048574)}
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 800012, Push 100096(top)
                Protocol next hop: 10.255.70.103
                Push 800012
                Indirect next hop: 87272e4 1048574
                State: <Active Int>
                Age: 1:31:53    Metric2: 2
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 1-Common L2 VC
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Indirect next hops: 1
                        Protocol next hop: 10.255.70.103 Metric: 2
                        Push 800012
                        Indirect next hop: 87272e4 1048574
```

```
                        Indirect path forwarding next hops: 1
                                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1
                        10.255.70.103/32 Originating RIB: inet.3
                          Metric: 2                     Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.31.1.6 via ge-3/1/0.0

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

abcd::10:255:71:52/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:34:07
                Task: IF
                AS path: I

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:34:07
                Task: IF
                AS path: I

ff02::2/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::2/128 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:34:08
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::d/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::d/128 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:34:08
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::16/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::16/128 -> {}
        *MLD    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
```

```
                Local AS:    69
                Age: 1:34:06
                Task: MLD
                Announcement bits (1): 0-KRT
                AS path: I

private.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.16385, selected
                State: <Active NoReadvrt Int>
                Age: 1:34:07
                Task: IF
                AS path: I

green.l2vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

10.255.70.103:1:3:1/96 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.70.103:1
                Next-hop reference count: 7
                Source: 10.255.70.103
                Protocol next hop: 10.255.70.103
                Indirect next hop: 2 no-forward
                State: <Secondary Active Int Ext>
                Local AS:    69 Peer AS:    69
                Age: 1:28:12    Metric2: 1
                Task: BGP_69.10.255.70.103+179
                Announcement bits (1): 0-green-l2vpn
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Label-base: 800008, range: 8
                Localpref: 100
                Router ID: 10.255.70.103
                Primary Routing Table bgp.l2vpn.0

10.255.71.52:1:1:1/96 (1 entry, 1 announced)
TSI:
Page 0 idx 0 Type 1 val 8699540
        *L2VPN  Preference: 170/-1
                Next-hop reference count: 5
                Protocol next hop: 10.255.71.52
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:34:03    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:Site-Down,
                mtu: 0
                Label-base: 800016, range: 8, status-vector: 0x9F

10.255.71.52:1:5:1/96 (1 entry, 1 announced)
TSI:
Page 0 idx 0 Type 1 val 8699528
        *L2VPN  Preference: 170/-101
                Next-hop reference count: 5
```

```
                    Protocol next hop: 10.255.71.52
                    Indirect next hop: 0 -
                    State: <Active Int Ext>
                    Age: 1:34:03    Metric2: 1
                    Task: green-l2vpn
                    Announcement bits (1): 1-BGP.0.0.0.0+179
                    AS path: I
                    Communities: Layer2-info: encaps:VPLS, control flags:, mtu: 0
                    Label-base: 800008, range: 8, status-vector: 0x9F

...

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

TSI:

10.245.255.63:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
                Protocol next hop: 10.245.255.63 Indirect next hop: 86af000 296
                State: <Active Int>
                Local AS:    99
                Age: 10:21
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 100000, MTU 1500, VLAN ID 512


55.0.0.0/24 (1 entry, 1 announced)
TSI:
KRT queued (pending) add
  55.0.0.0/24 -> {Push 300112}
        *BGP    Preference: 170/-101
                Next hop type: Router
                Address: 0x925c208
                Next-hop reference count: 2
                Source: 10.0.0.9
                Next hop: 10.0.0.9 via ge-1/2/0.15, selected
                Label operation: Push 300112
                Label TTL action: prop-ttl
                State: <Active Ext>
                Local AS:  7019 Peer AS: 13979
                Age: 1w0d 23:06:56
                AIGP: 25
                Task: BGP_13979.10.0.0.9+56732
                Announcement bits (1): 0-KRT
                AS path: 13979 7018 I
                Accepted
                Route Label: 300112
                Localpref: 100
                Router ID: 10.9.9.1
```

show route extensive (Access Route)

```
user@host> show route 13.160.0.102 extensive
inet.0: 39256 destinations, 39258 routes (39255 active, 0 holddown, 1 hidden)
13.160.0.102/32 (1 entry, 1 announced)
TSI:
```

```
KRT in-kernel 13.160.0.102/32 -> {13.160.0.2}
OSPF area : 0.0.0.0, LSA ID : 13.160.0.102, LSA type : Extern
        *Access Preference: 13
                Next-hop reference count: 78472
                Next hop: 13.160.0.2 via fe-0/0/0.0, selected
                State: <Active Int>
       Age: 12
                Task: RPD Unix Domain Server./var/run/rpd_serv.local
                Announcement bits (2): 0-KRT 1-OSPFv2
                AS path: I
```

show route extensive (BGP PIC Edge)

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

show route extensive (FRR and LFA)

```
user@host> show route 20:31:2:0 extensive
inet.0: 46 destinations, 49 routes (45 active, 0 holddown, 1 hidden)
20.31.2.0/24 (2 entries, 1 announced)
        State: FlashAll
TSI:
KRT in-kernel 20.31.2.0/24 -> {Push 299776, Push 299792}
        *RSVP   Preference: 7/1
                Next hop type: Router, Next hop index: 1048574
                Address: 0xbbbc010
                Next-hop reference count: 5
                Next hop: 10.31.1.2 via ge-2/1/8.0 weight 0x1, selected
                Label-switched-path europa-d-to-europa-e
```

```
                        Label operation: Push 299776
                        Label TTL action: prop-ttl
                        Session Id: 0x201
                        Next hop: 10.31.2.2 via ge-2/1/4.0 weight 0x4001
                        Label-switched-path europa-d-to-europa-e
                        Label operation: Push 299792
                        Label TTL action: prop-ttl
                        Session Id: 0x202
                        State: Active Int
                        Local AS:   100
                        Age: 5:31  Metric: 2
                        Task: RSVP
                        Announcement bits (1): 0-KRT
                        AS path: I
                 OSPF   Preference: 10
                        Next hop type: Router, Next hop index: 615
                        Address: 0xb9d78c4
                        Next-hop reference count: 7
                        Next hop: 10.31.1.2 via ge-2/1/8.0, selected
                        Session Id: 0x201
                        State: Int
                        Inactive reason: Route Preference
                        Local AS:   100
                        Age: 5:35  Metric: 3
                        Area: 0.0.0.0
                        Task: OSPF
                        AS path: I
```

show route extensive (Route Reflector)

```
user@host> show route extensive
1.0.0.0/8 (1 entry, 1 announced)

TSI:
KRT in-kernel 1.0.0.0/8 -> {indirect(40)}
        *BGP    Preference: 170/-101
                Source: 192.168.4.214
                Protocol next hop: 207.17.136.192 Indirect next hop: 84ac908 40
                State: <Active Int Ext>
                Local AS: 10458 Peer AS: 10458
                Age: 3:09       Metric: 0       Metric2: 0
                Task: BGP_10458.192.168.4.214+1033
                Announcement bits (2): 0-KRT 4-Resolve inet.0
                AS path: 3944 7777 I <Originator>
                Cluster list:  1.1.1.1
                Originator ID: 10.255.245.88
                Communities: 7777:7777
                Localpref: 100
                Router ID: 4.4.4.4
                Indirect next hops: 1
                        Protocol next hop: 207.17.136.192 Metric: 0
                        Indirect next hop: 84ac908 40
                        Indirect path forwarding next hops: 0
                                Next hop type: Discard
```

show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs)

```
user@host> show route label 299872 detail
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
299872 (1 entry, 1 announced)
        *LDP    Preference: 9
```

```
                    Next hop type: Flood
                    Next-hop reference count: 3
                    Address: 0x9097d90
                    Next hop: via vt-0/1/0.1
                    Next-hop index: 661
                    Label operation: Pop
                    Address: 0x9172130
                    Next hop: via so-0/0/3.0
                    Next-hop index: 654
                    Label operation: Swap 299872
                    State: **Active Int>
                    Local AS:  1001
                    Age: 8:20       Metric: 1
                    Task: LDP
                    Announcement bits (1): 0-KRT
                    AS path: I
                   FECs bound to route: P2MP root-addr 10.255.72.166, grp 232.1.1.1,
src 192.168.142.2
```

show route flow validation

Syntax
show route flow validation
<brief | detail>
<*ip-prefix*>
<table *table-name*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route flow validation
<brief | detail>
<*ip-prefix*>
<table *table-name*>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display flow route information.

Options
none—Display flow route information.

brief | detail—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

ip-prefix—(Optional) IP address for the flow route.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

table *table-name*—(Optional) Display flow route information for all routing tables whose name begins with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route flow validation inet** command).

Required Privilege Level
view

List of Sample Output
show route flow validation on page 569

Output Fields
Table 38 on page 568 lists the output fields for the **show route flow validation** command. Output fields are listed in the approximate order in which they appear.

Table 38: show route flow validation Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table (for example, inet.0).	All levels
prefix	Route address.	All levels
Active unicast route	Active route in the routing table.	All levels
Dependent flow destinations	Number of flows for which there are routes in the routing table.	All levels

Table 38: show route flow validation Output Fields *(continued)*

Field Name	Field Description	Level of Output
Origin	Source of the route flow.	All levels
Neighbor AS	Autonomous system identifier of the neighbor.	All levels
Flow destination	Number of entries and number of destinations that match the route flow.	All levels
Unicast best match	Destination that is the best match for the route flow.	All levels
Flags	Information about the route flow.	All levels

Sample Output

show route flow validation

```
user@host> show route flow validation
inet.0:
10.0.5.0/24Active unicast route
Dependent flow destinations: 1
Origin: 192.168.224.218, Neighbor AS: 65001
Flow destination (3 entries, 1 match origin)
Unicast best match: 10.0.5.0/24
Flags: SubtreeApex Consistent
```

show route forwarding-table

Syntax

show route forwarding-table
<detail | extensive | summary>
<all>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<label *name*>
<matching *matching*>
<multicast>
<table (default | *logical-system-name/routing-instance-name* | *routing-instance-name*)>
<vlan (all | *vlan-name*)>
<vpn *vpn*>

Syntax (MX Series Routers)

show route forwarding-table
<detail | extensive | summary>
<all>
<bridge-domain (all | *domain-name*)>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<label *name*>
<learning-vlan-id *learning-vlan-id*>
<matching *matching*>
<multicast>
<table (default | *logical-system-name/routing-instance-name* | *routing-instance-name*)>
<vlan (all | *vlan-name*)>
<vpn *vpn*>

Syntax (TX Matrix and TX Matrix Plus Routers)

show route forwarding-table
<detail | extensive | summary>
<all>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<matching *matching*>
<label *name*>
<lcc *number*>
<multicast>
<table *routing-instance-name*>
<vpn *vpn*>

Release Information

Command introduced before Junos OS Release 7.4.
Option **bridge-domain** introduced in Junos OS Release 7.5
Option **learning-vlan-id** introduced in Junos OS Release 8.4
Options **all** and **vlan** introduced in Junos OS Release 9.6.
Command introduced in Junos OS Release 11.3 for the QFX Series.

Description Display the Routing Engine's forwarding table, including the network-layer prefixes and their next hops. This command is used to help verify that the routing protocol process has relayed the correction information to the forwarding table. The Routing Engine constructs and maintains one or more routing tables. From the routing tables, the Routing Engine derives a table of active routes, called the forwarding table.

NOTE: The Routing Engine copies the forwarding table to the Packet Forwarding Engine, the part of the router that is responsible for forwarding packets. To display the entries in the Packet Forwarding Engine's forwarding table, use the **show pfe route** command.

Options **none**—Display the routes in the forwarding tables. By default, the **show route forwarding-table** command does not display information about private, or internal, forwarding tables.

detail | extensive | summary—(Optional) Display the specified level of output.

all—(Optional) Display routing table entries for all forwarding tables, including private, or internal, tables.

bridge-domain (all | *bridge-domain-name*)—(MX Series routers only) (Optional) Display route entries for all bridge domains or the specified bridge domain.

ccc *interface-name*—(Optional) Display route entries for the specified circuit cross-connect interface.

destination *destination-prefix*—(Optional) Destination prefix.

family *family*—(Optional) Display routing table entries for the specified family: **fibre-channel**, **fmembers**, **inet**, **inet6**, **iso**, **mpls**, **tnp**, **unix**, **vpls**, or **vlan-classification**.

interface-name *interface-name*—(Optional) Display routing table entries for the specified interface.

label *name*—(Optional) Display route entries for the specified label.

lcc *number*—(TX Matrix and TX matrix Plus routers only) (Optional) On a routing matrix composed of a TX Matrix router and T640 routers, display information for the specified T640 router (or line-card chassis) connected to the TX Matrix router. On a routing matrix composed of the TX Matrix Plus router and T1600 or T4000 routers, display information for the specified router (line-card chassis) connected to the TX Matrix Plus router.

Replace *number* with the following values depending on the LCC configuration:

- 0 through 3, when T640 routers are connected to a TX Matrix router in a routing matrix.
- 0 through 3, when T1600 routers are connected to a TX Matrix Plus router in a routing matrix.

- 0 through 7, when T1600 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.
- 0, 2, 4, or 6, when T4000 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.

learning-vlan-id ***learning-vlan-id***—(MX Series routers only) (Optional) Display learned information for all VLANs or for the specified VLAN.

matching ***matching***—(Optional) Display routing table entries matching the specified prefix or prefix length.

multicast—(Optional) Display routing table entries for multicast routes.

table (default | ***logical-system-name/routing-instance-name*** **|** ***routing-instance-name*****)**—(Optional) Display route entries for all the routing tables in the main routing instance or for the specified routing instance. If your device supports logical systems, you can also display route entries for the specified logical system and routing instance. To view the routing instances on your device, use the show route instance command.

vlan (all | ***vlan-name*****)**—(Optional) Display information for all VLANs or for the specified VLAN.

vpn ***vpn***—(Optional) Display routing table entries for a specified VPN.

Required Privilege Level view

List of Sample Output

show route forwarding-table on page 575
show route forwarding-table detail on page 576
show route forwarding-table destination extensive (Weights and Balances) on page 576
show route forwarding-table extensive on page 577
show route forwarding-table extensive (RPF) on page 578
show route forwarding-table family mpls on page 579
show route forwarding-table family vpls on page 579
show route forwarding-table family vpls extensive on page 579
show route forwarding-table table default on page 581
show route forwarding-table table
logical-system-name/routing-instance-name on page 582
show route forwarding-table vpn on page 582

Output Fields Table 39 on page 573 lists the output fields for the **show route forwarding-table** command. Output fields are listed in the approximate order in which they appear. Field names might be abbreviated (as shown in parentheses) when no level of output is specified, or when the **detail** keyword is used instead of the **extensive** keyword.

Table 39: show route forwarding-table Output Fields

Field Name	Field Description	Level of Output
Logical system	Name of the logical system. This field is displayed if you specify the **table** ***logical-system-name/routing-instance-name*** option on a device that is configured for and supports logical systems.	All levels
Routing table	Name of the routing table (for example, inet, inet6, mpls).	All levels
Address family	Address family (for example, **IP**, **IPv6**, **ISO**, **MPLS**, and **VPLS**).	All levels
Destination	Destination of the route.	**detail extensive**
Route Type (Type)	How the route was placed into the forwarding table. When the **detail** keyword is used, the route type might be abbreviated (as shown in parentheses): • **cloned** (**clon**)—(TCP or multicast only) Cloned route. • **destination** (**dest**)—Remote addresses directly reachable through an interface. • **destination down** (**iddn**)—Destination route for which the interface is unreachable. • **interface cloned** (**ifcl**)—Cloned route for which the interface is unreachable. • **route down** (**ifdn**)—Interface route for which the interface is unreachable. • **ignore** (**ignr**)—Ignore this route. • **interface** (**intf**)—Installed as a result of configuring an interface. • **permanent** (**perm**)—Routes installed by the kernel when the routing table is initialized. • **user**—Routes installed by the routing protocol process or as a result of the configuration.	All levels
Route Reference (RtRef)	Number of routes to reference.	**detail extensive**
Flags	Route type flags: • **none**—No flags are enabled. • **accounting**—Route has accounting enabled. • **cached**—Cache route. • **incoming-iface** ***interface-number***—Check against incoming interface. • **prefix load balance**—Load balancing is enabled for this prefix. • **rt nh decoupled**—Route has been decoupled from the next hop to the destination. • **sent to PFE**—Route has been sent to the Packet Forwarding Engine. • **static**—Static route.	**extensive**
Next hop	IP address of the next hop to the destination.	**detail extensive**

Table 39: show route forwarding-table Output Fields *(continued)*

Field Name	Field Description	Level of Output
Next hop Type (Type)	Next-hop type. When the **detail** keyword is used, the next-hop type might be abbreviated (as indicated in parentheses): • **broadcast (bcst)**—Broadcast. • **deny**—Deny. • **discard (dscd)** —Discard. • **hold**—Next hop is waiting to be resolved into a unicast or multicast type. • **indexed (idxd)**—Indexed next hop. • **indirect (indr)**—Indirect next hop. • **local (locl)**—Local address on an interface. • **routed multicast (mcrt)**—Regular multicast next hop. • **multicast (mcst)**—Wire multicast next hop (limited to the LAN). • **multicast discard (mdsc)**—Multicast discard. • **multicast group (mgrp)**—Multicast group member. • **receive (recv)**—Receive. • **reject (rjct)**—Discard. An ICMP unreachable message was sent. • **resolve (rslv)**—Resolving the next hop. • **unicast (ucst)**—Unicast. • **unilist (ulst)**—List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.	**detail extensive**
Index	Software index of the next hop that is used to route the traffic for a given prefix.	**detail extensive** none
Route interface-index	Logical interface index from which the route is learned. For example, for interface routes, this is the logical interface index of the route itself. For static routes, this field is zero. For routes learned through routing protocols, this is the logical interface index from which the route is learned.	**extensive**
Reference (NhRef)	Number of routes that refer to this next hop.	**detail extensive** none
Next-hop interface (Netif)	Interface used to reach the next hop.	**detail extensive** none
Weight	Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible (see the **Balance** field description).	**extensive**
Balance	Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a router is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.	**extensive**
RPF interface	List of interfaces from which the prefix can be accepted. Reverse path forwarding (RPF) information is displayed only when **rpf-check** is configured on the interface.	**extensive**

Sample Output

show route forwarding-table

```
user@host> show route forwarding-table
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     4
0.0.0.0/32         perm     0                    dscd    44     1
1.1.1.0/24         ifdn     0                    rslv   608     1 ge-2/0/1.0
1.1.1.0/32         iddn     0 1.1.1.0            recv   606     1 ge-2/0/1.0
1.1.1.1/32         user     0                    rjct    46     4
1.1.1.1/32         intf     0 1.1.1.1            locl   607     2
1.1.1.1/32         iddn     0 1.1.1.1            locl   607     2
1.1.1.255/32       iddn     0 ff:ff:ff:ff:ff:ff  bcst   605     1 ge-2/0/1.0
10.0.0.0/24        intf     0                    rslv   616     1 ge-2/0/0.0
10.0.0.0/32        dest     0 10.0.0.0           recv   614     1 ge-2/0/0.0
10.0.0.1/32        intf     0 10.0.0.1           locl   615     2
10.0.0.1/32        dest     0 10.0.0.1           locl   615     2
10.0.0.255/32      dest     0 10.0.0.255         bcst   613     1 ge-2/0/0.0
10.1.1.0/24        ifdn     0                    rslv   612     1 ge-2/0/1.0
10.1.1.0/32        iddn     0 10.1.1.0           recv   610     1 ge-2/0/1.0
10.1.1.1/32        user     0                    rjct    46     4
10.1.1.1/32        intf     0 10.1.1.1           locl   611     2
10.1.1.1/32        iddn     0 10.1.1.1           locl   611     2
10.1.1.255/32      iddn     0 ff:ff:ff:ff:ff:ff  bcst   609     1 ge-2/0/1.0
10.206.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0
10.209.0.0/16      user     1 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.0.0/18      intf     0                    rslv   418     1 fxp0.0
10.209.0.0/32      dest     0 10.209.0.0         recv   416     1 fxp0.0
10.209.2.131/32    intf     0 10.209.2.131       locl   417     2
10.209.2.131/32    dest     0 10.209.2.131       locl   417     2
10.209.17.55/32    dest     0 0:30:48:5b:78:d2   ucst   435     1 fxp0.0
10.209.63.42/32    dest     0 0:23:7d:58:92:ca   ucst   434     1 fxp0.0
10.209.63.254/32   dest     0 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.63.255/32   dest     0 10.209.63.255      bcst   415     1 fxp0.0
10.227.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct    27     1
47.0005.80ff.f800.0000.0108.0003.0102.5524.5220.00
intf     0                  locl    28     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct     6     1
ff00::/8           perm     0                   mdsc     4     1
ff02::1/128        perm     0 ff02::1           mcst     3     1

Routing table: ccc
MPLS:
Interface.Label    Type RtRef Next hop           Type Index NhRef Netif
```

```
default            perm     0                   rjct 16     1
100004(top)fe-0/0/1.0
```

show route forwarding-table detail

```
user@host> show route forwarding-table detail
Routing table: inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            user     2 0:90:69:8e:b1:1b   ucst   132     4 fxp0.0
default            perm     0                    rjct    14     1
10.1.1.0/24        intf     0 ff.3.0.21          ucst   322     1 so-5/3/0.0
10.1.1.0/32        dest     0 10.1.1.0           recv   324     1 so-5/3/0.0
10.1.1.1/32        intf     0 10.1.1.1           locl   321     1
10.1.1.255/32      dest     0 10.1.1.255         bcst   323     1 so-5/3/0.0
10.21.21.0/24      intf     0 ff.3.0.21          ucst   326     1 so-5/3/0.0
10.21.21.0/32      dest     0 10.21.21.0         recv   328     1 so-5/3/0.0
10.21.21.1/32      intf     0 10.21.21.1         locl   325     1
10.21.21.255/32    dest     0 10.21.21.255       bcst   327     1 so-5/3/0.0
127.0.0.1/32       intf     0 127.0.0.1          locl   320     1
172.17.28.19/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0
172.17.28.44/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0

...

Routing table: private1__.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     1
10.0.0.0/8         intf     0                    rslv   136     1 fxp1.0
10.0.0.0/32        dest     0 10.0.0.0           recv   134     1 fxp1.0
10.0.0.4/32        intf     0 10.0.0.4           locl   135     2
10.0.0.4/32        dest     0 10.0.0.4           locl   135     2

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    38     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    22     1
ff00::/8           perm     0                    mdsc    21     1
ff02::1/128        perm     0 ff02::1            mcst    17     1

...

Routing table: mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct 28     1
```

show route forwarding-table destination extensive (Weights and Balances)

```
user@host> show route forwarding-table destination 3.4.2.1 extensive
Routing table: inet [Index 0]
Internet:
```

```
Destination:  3.4.2.1/32
  Route type: user
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: unilist                Index: 262143   Reference: 1
  Nexthop: 4.4.4.4
  Next-hop type: unicast                Index: 335      Reference: 2
  Next-hop interface: so-1/1/0.0        Weight: 22      Balance: 3
  Nexthop: 145.12.1.2
  Next-hop type: unicast                Index: 337      Reference: 2
  Next-hop interface: so-0/1/2.0        Weight: 33      Balance: 33
```

show route forwarding-table extensive

```
user@host> show route forwarding-table extensive
Routing table: inet [Index 0]
Internet:

Destination:  default
  Route type: user
  Route reference: 2                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 0:90:69:8e:b1:1b
  Next-hop type: unicast                Index: 132      Reference: 4
  Next-hop interface: fxp0.0

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: none
  Next-hop type: reject                 Index: 14       Reference: 1

Destination:  127.0.0.1/32
  Route type: interface
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 127.0.0.1
  Next-hop type: local                  Index: 320      Reference: 1

...

Routing table: private1__.inet [Index 1]
Internet:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 46       Reference: 1

Destination:  10.0.0.0/8
  Route type: interface
  Route reference: 0                    Route interface-index: 3
  Flags: sent to PFE
  Next-hop type: resolve                Index: 136      Reference: 1
  Next-hop interface: fxp1.0

...

Routing table: iso [Index 0]
ISO:
```

```
Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 38       Reference: 1

Routing table: inet6 [Index 0]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 22       Reference: 1

Destination:  ff00::/8
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: multicast discard      Index: 21       Reference: 1

...

Routing table: private1__.inet6 [Index 1]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 54       Reference: 1

Destination:  fe80::2a0:a5ff:fe3d:375/128
  Route type: interface
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: fe80::2a0:a5ff:fe3d:375
  Next-hop type: local                  Index: 75       Reference: 1

...
```

show route forwarding-table extensive (RPF)

The next example is based on the following configuration, which enables an RPF check on all routes that are learned from this interface, including the interface route:

```
so-1/1/0 {
    unit 0 {
        family inet {
            rpf-check;
            address 15.95.1.2/30;
        }
    }
}

user@host> show route forwarding-table extensive
Routing table: inet [Index 0]
Internet:
...
...
```

```
Destination:  15.95.1.3/32
  Route type: destination
  Route reference: 0                   Route interface-index: 67
  Flags: sent to PFE
  Nexthop: 15.95.1.3
  Next-hop type: broadcast             Index: 328      Reference: 1
  Next-hop interface: so-1/1/0.0
  RPF interface: so-1/1/0.0
```

show route forwarding-table family mpls

```
user@host> show route forwarding-table family mpls
Routing table: mpls
MPLS:
Destination        Type RtRef Next hop          Type Index NhRef Netif
default            perm     0                   rjct    19     1
0                  user     0                   recv    18     3
1                  user     0                   recv    18     3
2                  user     0                   recv    18     3
100000             user     0 10.31.1.6         swap  100001     fe-1/1/0.0
800002             user     0                   Pop              vt-0/3/0.32770

vt-0/3/0.32770 (VPLS)
                   user     0                   indr   351     4
                                                Push 800000, Push 100002(top)
so-0/0/0.0
```

show route forwarding-table family vpls

```
user@host> show route forwarding-table family vpls
Routing table: green.vpls
VPLS:
Destination        Type RtRef Next hop          Type Index NhRef Netif
default            dynm     0                   flood   353     1
default            perm     0                   rjct    298     1
fe-0/1/0.0         dynm     0                   flood   355     1
00:90:69:0c:20:1f/48             <<<<<Remote CE

                   dynm     0                   indr   351     4
                                                Push 800000, Push 100002(top)
so-0/0/0.0
00:90:69:85:b0:1f/48             <<<<<<Local CE

                   dynm     0                   ucst   354     2 fe-0/1/0.0
```

show route forwarding-table family vpls extensive

```
user@host> show route forwarding-table family vpls extensive
Routing table: green.vpls [Index 2]
VPLS:

Destination:  default
  Route type: dynamic
  Route reference: 0                   Route interface-index: 72
  Flags: sent to PFE
  Next-hop type: flood                 Index: 289      Reference: 1
  Next-hop type: unicast               Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0
  Next-hop type: unicast               Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0

Destination:  default
```

```
  Route type: permanent
  Route reference: 0                   Route interface-index: 0
  Flags: none
  Next-hop type: discard               Index: 341      Reference: 1

Destination:  fe-0/1/2.0
  Route type: dynamic
  Route reference: 0                   Route interface-index: 69
  Flags: sent to PFE
  Next-hop type: flood                 Index: 293      Reference: 1
  Next-hop type: indirect              Index: 363      Reference: 4
  Next-hop type: Push 800016
  Next-hop interface: at-1/0/1.0
  Next-hop type: indirect              Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
  Next-hop type: unicast               Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0

Destination:  fe-0/1/3.0
  Route type: dynamic
  Route reference: 0                   Route interface-index: 70
  Flags: sent to PFE
  Next-hop type: flood                 Index: 292      Reference: 1
  Next-hop type: indirect              Index: 363      Reference: 4
  Next-hop type: Push 800016
  Next-hop interface: at-1/0/1.0
  Next-hop type: indirect              Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
  Next-hop type: unicast               Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0

Destination:  10:00:00:01:01:01/48
  Route type: dynamic
  Route reference: 0                   Route interface-index: 70
  Flags: sent to PFE, prefix load balance
  Next-hop type: unicast               Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0
  Route used as destination:
    Packet count:         6640    Byte count:      675786
  Route used as source
    Packet count:         6894    Byte count:      696424

Destination:  10:00:00:01:01:04/48
  Route type: dynamic
  Route reference: 0                   Route interface-index: 69
  Flags: sent to PFE, prefix load balance
  Next-hop type: unicast               Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0
  Route used as destination:
    Packet count:           96    Byte count:        8079
  Route used as source:
    Packet count:          296    Byte count:       24955

Destination:  10:00:00:01:03:05/48
  Route type: dynamic
  Route reference: 0                   Route interface-index: 74
  Flags: sent to PFE, prefix load balance
```

```
  Next-hop type: indirect              Index: 301      Reference: 5
  Next hop: 10.31.3.2
  Next-hop type: Push 800000
  Next-hop interface: fe-0/1/1.0
```

show route forwarding-table table default

```
user@host> show route forwarding-table table default
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     2
0.0.0.0/32         perm     0                    dscd    34     1
10.0.60.0/30       user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.60.12/30      intf     0                    rslv   688     1 fe-0/1/3.0
10.0.60.12/32      dest     0 10.0.60.12         recv   686     1 fe-0/1/3.0
10.0.60.13/32      dest     0 0:5:85:8b:bc:22    ucst   713     5 fe-0/1/3.0
10.0.60.14/32      intf     0 10.0.60.14         locl   687     2
10.0.60.14/32      dest     0 10.0.60.14         locl   687     2
10.0.60.15/32      dest     0 10.0.60.15         bcst   685     1 fe-0/1/3.0
10.0.67.12/30      user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.80.0/30       ifdn     0 ff.3.0.21          ucst   676     1 so-0/0/1.0
10.0.80.0/32       dest     0 10.0.80.0          recv   678     1 so-0/0/1.0
10.0.80.2/32       user     0                    rjct    36     2
10.0.80.2/32       intf     0 10.0.80.2          locl   675     1
10.0.80.3/32       dest     0 10.0.80.3          bcst   677     1 so-0/0/1.0
10.0.90.12/30      intf     0                    rslv   684     1 fe-0/1/0.0
10.0.90.12/32      dest     0 10.0.90.12         recv   682     1 fe-0/1/0.0
10.0.90.14/32      intf     0 10.0.90.14         locl   683     2
10.0.90.14/32      dest     0 10.0.90.14         locl   683     2
10.0.90.15/32      dest     0 10.0.90.15         bcst   681     1 fe-0/1/0.0
10.5.0.0/16        user     0 192.168.187.126    ucst   324    15 fxp0.0
10.10.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.13.10.0/23      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.84.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.150.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.157.64.0/19     user     0 192.168.187.126    ucst   324    15 fxp0.0
10.209.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0

...

Routing table: default.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    60     1

Routing table: default.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    44     1
::/128             perm     0                    dscd    42     1
ff00::/8           perm     0                    mdsc    43     1
ff02::1/128        perm     0 ff02::1            mcst    39     1

Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd    50     1
```

show route forwarding-table table logical-system-name/routing-instance-name

```
user@host> show route forwarding-table table R4/vpn-red
Logical system: R4
Routing table: vpn-red.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   563     1
0.0.0.0/32         perm     0                    dscd   561     2
1.0.0.1/32         user     0                    dscd   561     2
2.0.2.0/24         intf     0                    rslv   771     1 ge-1/2/0.3
2.0.2.0/32         dest     0 2.0.2.0            recv   769     1 ge-1/2/0.3
2.0.2.1/32         intf     0 2.0.2.1            locl   770     2
2.0.2.1/32         dest     0 2.0.2.1            locl   770     2
2.0.2.2/32         dest     0 0.4.80.3.0.1b.c0.d5.e4.bd.0.1b.c0.d5.e4.bc.8.0
                                                 ucst   789     1 ge-1/2/0.3
2.0.2.255/32       dest     0 2.0.2.255          bcst   768     1 ge-1/2/0.3
224.0.0.0/4        perm     1                    mdsc   562     1
224.0.0.1/32       perm     0 224.0.0.1          mcst   558     1
255.255.255.255/32 perm     0                    bcst   559     1

Logical system: R4
Routing table: vpn-red.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   608     1

Logical system: R4
Routing table: vpn-red.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   708     1
::/128             perm     0                    dscd   706     1
ff00::/8           perm     0                    mdsc   707     1
ff02::1/128        perm     0 ff02::1            mcst   704     1

Logical system: R4
Routing table: vpn-red.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd   638
```

show route forwarding-table vpn

```
user@host> show route forwarding-table vpn VPN-A
      Routing table:: VPN-A.inet
Internet:
      Destination        Type RtRef Nexthop           Type Index NhRef Netif
      default            perm     0                   rjct     4     4
      10.39.10.20/30     intf     0 ff.3.0.21         ucst    40     1
      so-0/0/0.0
      10.39.10.21/32     intf     0 10.39.10.21       locl    36     1
      10.255.14.172/32   user     0                   ucst    69     2
      so-0/0/0.0
      10.255.14.175/32   user     0                   indr    81     3
                                                      Push 100004, Push
      100004(top) so-1/0/0.0
      224.0.0.0/4        perm     2                   mdsc     5     3
      224.0.0.1/32       perm     0 224.0.0.1         mcst     1     8
```

```
224.0.0.5/32       user     1 224.0.0.5         mcst     1     8
255.255.255.255/32 perm     0                   bcst     2     3
```

show route hidden

Syntax show route hidden
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Release Information Command introduced before Junos OS Release 7.4.

Description Display only hidden route information. A hidden route is unusable, even if it is the best path.

Options **brief | detail | extensive | terse**—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level view

Related Documentation
- *Understanding Hidden Routes*

List of Sample Output
show route hidden on page 584
show route hidden detail on page 585
show route hidden extensive on page 585
show route hidden terse on page 585

Output Fields For information about output fields, see the output field table for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route hidden

```
user@host> show route hidden
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
127.0.0.1/32        [Direct/0] 04:26:38
                    > via lo0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.5.5.5/32         [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: 100 I
                      Unusable
10.12.1.0/24        [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: 100 I
                      Unusable
```

```
10.12.80.4/30       [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: I
                      Unusable
...
```

show route hidden detail

```
user@host> show route hidden detail

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
127.0.0.1/32 (1 entry, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Hidden Martian Int>
                Local AS:     1
                Age: 4:27:37
                Task: IF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete

10.5.5.5/32 (1 entry, 0 announced)
         BGP    Preference: 170/-101
                Route Distinguisher: 10.4.4.4:4
                Next hop type: Unusable
                Next-hop reference count: 6
                State: <Secondary Hidden Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 3:45:09
                Task: BGP_1.10.4.4.4+2493
                AS path: 100 I
                Communities: target:1:999
                VPN Label: 100064
                Localpref: 100
                Router ID: 10.4.4.4
                Primary Routing Table bgp.l3vpn.0

...
```

show route hidden extensive

The output for the **show route hidden extensive** command is identical to that of the **show route hidden detail** command. For sample output, see show route hidden detail on page 585.

show route hidden terse

```
user@host> show route hidden terse

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  127.0.0.1/32       D   0                     >lo0.0
```

```
private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.5.5.5/32        B 170        100             Unusable        100 I
  10.12.1.0/24       B 170        100             Unusable        100 I
  10.12.80.4/30      B 170        100             Unusable        I

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.4.4.4:4:10.5.5.5/32
                     B 170        100             Unusable        100 I
  10.4.4.4:4:10.12.1.0/24
                     B 170        100             Unusable        100 I
  10.4.4.4:4:10.12.80.4/30
                     B 170        100             Unusable        I

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route inactive-path

Syntax
show route inactive-path
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route inactive-path
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display routes for destinations that have no active route. An inactive route is a route that was not selected as the best path.

Options
none—Display all inactive routes.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route inactive-path on page 587
show route inactive-path detail on page 588
show route inactive-path extensive on page 589
show route inactive-path terse on page 589

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route inactive-path

```
user@host> show route inactive-path

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.12.100.12/30     [OSPF/10] 03:57:28, metric 1
                    > via so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/8          [Direct/0] 04:39:56
                    > via fxp1.0
```

```
red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.12.80.0/30       [BGP/170] 04:38:17, localpref 100
                      AS path: 100 I
                    > to 10.12.80.1 via ge-6/3/2.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route inactive-path detail

```
user@host> show route inactive-path detail

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete

10.12.100.12/30 (2 entries, 1 announced)
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Local AS:     1
                Age: 3:58:24    Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

10.0.0.0/8 (2 entries, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp1.0, selected
                State: <NotBest Int>
                Inactive reason: No difference
                Age: 4:40:52
                Task: IF
                AS path: I

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete

10.12.80.0/30 (2 entries, 1 announced)
         BGP    Preference: 170/-101
                Next-hop reference count: 6
                Source: 10.12.80.1
```

```
Next hop: 10.12.80.1 via ge-6/3/2.0, selected
State: <Ext>
Inactive reason: Route Preference
Peer AS:   100
Age: 4:39:13
Task: BGP_100.10.12.80.1+179
AS path: 100 I
Localpref: 100
Router ID: 10.0.0.0
```

show route inactive-path extensive

The output for the **show route inactive-path extensive** command is identical to that of the **show route inactive-path detail** command. For sample output, see show route inactive-path detail on page 588.

show route inactive-path terse

```
user@host> show route inactive-path terse

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.12.100.12/30    O  10          1            >so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.0.0.0/8         D   0                     >fxp1.0

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.12.80.0/30      B 170        100            >10.12.80.1      100 I

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route inactive-prefix

Syntax
show route inactive-prefix
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route inactive-prefix
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display inactive route destinations in each routing table.

Options
none—Display all inactive route destination.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route inactive-prefix on page 590
show route inactive-prefix detail on page 590
show route inactive-prefix extensive on page 591
show route inactive-prefix terse on page 591

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route inactive-prefix

```
user@host> show route inactive-prefix

inet.0: 14 destinations, 14 routes (13 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

127.0.0.1/32        [Direct/0] 00:04:54
                    > via lo0.0
```

show route inactive-prefix detail

```
user@host> show route inactive-prefix detail

inet.0: 14 destinations, 14 routes (13 active, 0 holddown, 1 hidden)
127.0.0.1/32 (1 entry, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
```

```
                    Next-hop reference count: 1
                    Next hop: via lo0.0, selected
                    State: <Hidden Martian Int>
                    Age: 4:51
                    Task: IF
                    AS path: I00:04:54
                        > via lo0.0
```

show route inactive-prefix extensive

The output for the **show route inactive-prefix extensive** command is identical to that of the **show route inactive-path detail** command. For sample output, see show route inactive-prefix detail on page 590.

show route inactive-prefix terse

```
user@host> show route inactive-prefix terse

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  127.0.0.1/32       D   0                     >lo0.0
```

show route instance

Syntax

show route instance
<brief | detail | summary>
<*instance-name*>
<logical-system (all | *logical-system-name*)>
<operational>

Syntax (EX Series Switches and QFX Series)

show route instance
<brief | detail | summary>
<*instance-name*>
<operational>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.

Description

Display routing instance information.

Options

none—(Same as **brief**) Display standard information about all routing instances.

brief | detail | summary—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**. (These options are not available with the **operational** keyword.)

instance-name—(Optional) Display information for all routing instances whose name begins with this string (for example, **cust1**, **cust11**, and **cust111** are all displayed when you run the **show route instance cust1** command).

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

operational—(Optional) Display operational routing instances.

Required Privilege Level

view

List of Sample Output

show route instance on page 593
show route instance detail (Graceful Restart Complete) on page 594
show route instance detail (Graceful Restart Incomplete) on page 595
show route instance detail (VPLS Routing Instance) on page 597
show route instance operational on page 598
show route instance summary on page 598

Output Fields

Table 40 on page 592 lists the output fields for the **show route instance** command. Output fields are listed in the approximate order in which they appear.

Table 40: show route instance Output Fields

Field Name	Field Description	Level of Output
Instance or *instance-name*	Name of the routing instance.	All levels

Table 40: show route instance Output Fields *(continued)*

Field Name	Field Description	Level of Output
Operational Routing Instances	(**operational** keyword only) Names of all operational routing instances.	—
Type	Type of routing instance: **forwarding**, **l2vpn**, **no-forwarding**, **vpls**, **virtual-router**, or **vrf**.	All levels
State	State of the routing instance: **active** or **inactive**.	**brief detail** none
Interfaces	Name of interfaces belonging to this routing instance.	**brief detail** none
Restart State	Status of graceful restart for this instance: **Pending** or **Complete**.	**detail**
Path selection timeout	Maximum amount of time, in seconds, remaining until graceful restart is declared complete. The default is **300**.	**detail**
Tables	Tables (and number of routes) associated with this routing instance.	**brief detail** none
Route-distinguisher	Unique route distinguisher associated with this routing instance.	**detail**
Vrf-import	VPN routing and forwarding instance import policy name.	**detail**
Vrf-export	VPN routing and forwarding instance export policy name.	**detail**
Vrf-import-target	VPN routing and forwarding instance import target community name.	**detail**
Vrf-export-target	VPN routing and forwarding instance export target community name.	**detail**
Fast-reroute-priority	Fast reroute priority setting for a VPLS routing instance: **high**, **medium**, or **low**. The default is **low**.	**detail**
Restart State	Restart state: • **Pending:*protocol-name***—List of protocols that have not yet completed graceful restart for this routing table. • **Complete**—All protocols have restarted for this routing table.	**detail**
Primary rib	Primary table for this routing instance.	**brief** none **summary**
Active/holddown/hidden	Number of active, hold-down, and hidden routes.	All levels

Sample Output

show route instance

```
user@host> show route instance
Instance             Type
        Primary RIB                                  Active/holddown/hidden
master               forwarding
        inet.0                                       16/0/1
```

```
        iso.0                                              1/0/0
        mpls.0                                             0/0/0
        inet6.0                                            2/0/0
        l2circuit.0                                        0/0/0
__juniper_private1__ forwarding
        __juniper_private1__.inet.0                        12/0/0
        __juniper_private1__.inet6.0                       1/0/0
```

show route instance detail (Graceful Restart Complete)

```
user@host> show route instance detail
master:
  Router ID: 10.255.14.176
  Type: forwarding        State: Active
  Restart State: Complete Path selection timeout: 300
  Tables:
    inet.0                : 17 routes (15 active, 0 holddown, 1 hidden)
    Restart Complete
    inet.3                : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    iso.0                 : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
    mpls.0                : 19 routes (19 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l3vpn.0           : 10 routes (10 active, 0 holddown, 0 hidden)
    Restart Complete
    inet6.0               : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l2vpn.0           : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
BGP-INET:
  Router ID: 10.69.103.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.103
  Route-distinguisher: 10.255.14.176:103
  Vrf-import: [ BGP-INET-import ]
  Vrf-export: [ BGP-INET-export ]
  Tables:
    BGP-INET.inet.0       : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
BGP-L:
  Router ID: 10.69.104.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.104
  Route-distinguisher: 10.255.14.176:104
  Vrf-import: [ BGP-L-import ]
  Vrf-export: [ BGP-L-export ]
  Tables:
    BGP-L.inet.0          : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
    BGP-L.mpls.0          : 3 routes (3 active, 0 holddown, 0 hidden)
    Restart Complete
L2VPN:
  Router ID: 0.0.0.0
  Type: l2vpn             State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
```

```
    t3-0/0/0.512
  Route-distinguisher: 10.255.14.176:512
  Vrf-import: [ L2VPN-import ]
  Vrf-export: [ L2VPN-export ]
  Tables:
    L2VPN.l2vpn.0          : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
LDP:
  Router ID: 10.69.105.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.105
  Route-distinguisher: 10.255.14.176:105
  Vrf-import: [ LDP-import ]
  Vrf-export: [ LDP-export ]
  Tables:
    LDP.inet.0             : 5 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
OSPF:
  Router ID: 10.69.101.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.101
  Route-distinguisher: 10.255.14.176:101
  Vrf-import: [ OSPF-import ]
  Vrf-export: [ OSPF-export ]
  Vrf-import-target: [ target:11111
  Tables:
    OSPF.inet.0            : 8 routes (7 active, 0 holddown, 0 hidden)
    Restart Complete
RIP:
  Router ID: 10.69.102.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.102
  Route-distinguisher: 10.255.14.176:102
  Vrf-import: [ RIP-import ]
  Vrf-export: [ RIP-export ]
  Tables:
    RIP.inet.0             : 6 routes (6 active, 0 holddown, 0 hidden)
    Restart Complete
STATIC:
  Router ID: 10.69.100.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.100
  Route-distinguisher: 10.255.14.176:100
  Vrf-import: [ STATIC-import ]
  Vrf-export: [ STATIC-export ]
  Tables:
    STATIC.inet.0          : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
```

show route instance detail (Graceful Restart Incomplete)

```
user@host> show route instance detail
```

```
master:
  Router ID: 10.255.14.176
  Type: forwarding        State: Active
  Restart State: Pending  Path selection timeout: 300
  Tables:
    inet.0                : 17 routes (15 active, 1 holddown, 1 hidden)
    Restart Pending: OSPF LDP
    inet.3                : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: OSPF LDP
    iso.0                 : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
    mpls.0                : 23 routes (23 active, 0 holddown, 0 hidden)
    Restart Pending: LDP VPN
    bgp.l3vpn.0           : 10 routes (10 active, 0 holddown, 0 hidden)
    Restart Pending: BGP VPN
    inet6.0               : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l2vpn.0           : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Pending: BGP VPN
BGP-INET:
  Router ID: 10.69.103.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.103
  Route-distinguisher: 10.255.14.176:103
  Vrf-import: [ BGP-INET-import ]
  Vrf-export: [ BGP-INET-export ]
  Tables:
    BGP-INET.inet.0       : 6 routes (5 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
BGP-L:
  Router ID: 10.69.104.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.104
  Route-distinguisher: 10.255.14.176:104
  Vrf-import: [ BGP-L-import ]
  Vrf-export: [ BGP-L-export ]
  Tables:
    BGP-L.inet.0          : 6 routes (5 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
    BGP-L.mpls.0          : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
L2VPN:
  Router ID: 0.0.0.0
  Type: l2vpn             State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.512
  Route-distinguisher: 10.255.14.176:512
  Vrf-import: [ L2VPN-import ]
  Vrf-export: [ L2VPN-export ]
  Tables:
    L2VPN.l2vpn.0         : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: VPN L2VPN
LDP:
  Router ID: 10.69.105.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
```

```
  Interfaces:
    t3-0/0/0.105
  Route-distinguisher: 10.255.14.176:105
  Vrf-import: [ LDP-import ]
  Vrf-export: [ LDP-export ]
  Tables:
    LDP.inet.0              : 5 routes (4 active, 1 holddown, 0 hidden)
    Restart Pending: OSPF LDP VPN
OSPF:
  Router ID: 10.69.101.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.101
  Route-distinguisher: 10.255.14.176:101
  Vrf-import: [ OSPF-import ]
  Vrf-export: [ OSPF-export ]
  Tables:
    OSPF.inet.0             : 8 routes (7 active, 1 holddown, 0 hidden)
    Restart Pending: OSPF VPN
RIP:
  Router ID: 10.69.102.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.102
  Route-distinguisher: 10.255.14.176:102
  Vrf-import: [ RIP-import ]
  Vrf-export: [ RIP-export ]
  Tables:
    RIP.inet.0              : 8 routes (6 active, 2 holddown, 0 hidden)
    Restart Pending: RIP VPN
STATIC:
  Router ID: 10.69.100.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.100
  Route-distinguisher: 10.255.14.176:100
  Vrf-import: [ STATIC-import ]
  Vrf-export: [ STATIC-export ]
  Tables:
    STATIC.inet.0           : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
```

show route instance detail (VPLS Routing Instance)

```
user@host> show route instance detail test-vpls
test-vpls:
  Router ID: 0.0.0.0
  Type: vpls              State: Active
  Interfaces:
    lsi.1048833
    lsi.1048832
    fe-0/1/0.513
  Route-distinguisher: 10.255.37.65:1
  Vrf-import: [ __vrf-import-test-vpls-internal__ ]
  Vrf-export: [ __vrf-export-test-vpls-internal__ ]
  Vrf-import-target: [ target:300:1 ]
  Vrf-export-target: [ target:300:1 ]
  Fast-reroute-priority: high
```

```
  Tables:
    test-vpls.l2vpn.0          : 3 routes (3 active, 0 holddown, 0 hidden)
```

show route instance operational

```
user@host> show route instance operational
Operational Routing Instances:

master
default
```

show route instance summary

```
user@host> show route instance summary
Instance             Type            Primary rib           Active/holddown/hidden
master               forwarding
                                     inet.0                15/0/1
                                     iso.0                 1/0/0
                                     mpls.0                35/0/0
                                     l3vpn.0               0/0/0
                                     inet6.0               2/0/0
                                     l2vpn.0               0/0/0
                                     l2circuit.0           0/0/0
BGP-INET             vrf
                                     BGP-INET.inet.0       5/0/0
                                     BGP-INET.iso.0        0/0/0
                                     BGP-INET.inet6.0      0/0/0
BGP-L                vrf
                                     BGP-L.inet.0          5/0/0
                                     BGP-L.iso.0           0/0/0
                                     BGP-L.mpls.0          4/0/0
                                     BGP-L.inet6.0         0/0/0
L2VPN                l2vpn
                                     L2VPN.inet.0          0/0/0
                                     L2VPN.iso.0           0/0/0
                                     L2VPN.inet6.0         0/0/0
                                     L2VPN.l2vpn.0         2/0/0
LDP                  vrf
                                     LDP.inet.0            4/0/0
                                     LDP.iso.0             0/0/0
                                     LDP.mpls.0            0/0/0
                                     LDP.inet6.0           0/0/0
                                     LDP.l2circuit.0       0/0/0
OSPF                 vrf
                                     OSPF.inet.0           7/0/0
                                     OSPF.iso.0            0/0/0
                                     OSPF.inet6.0          0/0/0
RIP                  vrf
                                     RIP.inet.0            6/0/0
                                     RIP.iso.0             0/0/0
                                     RIP.inet6.0           0/0/0
STATIC               vrf
                                     STATIC.inet.0         4/0/0
                                     STATIC.iso.0          0/0/0
                                     STATIC.inet6.0        0/0/0
```

show route next-hop

Syntax
show route next-hop *next-hop*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route next-hop *next-hop*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the entries in the routing table that are being sent to the specified next-hop address.

Options
brief | detail | extensive | terse—(Optional) Display the specified level of ouput.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

next-hop—Next-hop address.

Required Privilege Level
view

List of Sample Output
show route next-hop on page 599
show route next-hop detail on page 600
show route next-hop extensive on page 602
show route next-hop terse on page 603

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route next-hop

```
user@host> show route next-hop 192.168.71.254

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.10.0.0/16       *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
10.209.0.0/16      *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
172.16.0.0/12      *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
192.168.0.0/16     *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
192.168.102.0/23   *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
```

```
207.17.136.0/24    *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
207.17.136.192/32  *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop detail

```
user@host> show route next-hop 192.168.71.254 detail

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
Restart Complete
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

172.16.0.0/12 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

192.168.0.0/16 (1 entry, 1 announced)
```

```
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

192.168.102.0/23 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

207.17.136.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

207.17.136.192/32 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop extensive

```
user@host> show route next-hop 192.168.71.254 extensive

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.10.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.209.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

172.16.0.0/12 (1 entry, 1 announced)
TSI:
KRT in-kernel 172.16.0.0/12 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

192.168.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

192.168.102.0/23 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.102.0/23 -> {192.168.71.254}
        *Static Preference: 5
```

```
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

207.17.136.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.0/24 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

207.17.136.192/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.192/32 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 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)

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

green.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

red.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop terse

```
user@host> show route next-hop 192.168.71.254 terse

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.10.0.0/16       S   5                      >192.168.71.254
* 10.209.0.0/16      S   5                      >192.168.71.254
* 172.16.0.0/12      S   5                      >192.168.71.254
```

```
* 192.168.0.0/16     S   5                       >192.168.71.254
* 192.168.102.0/23   S   5                       >192.168.71.254
* 207.17.136.0/24    S   5                       >192.168.71.254
* 207.17.136.192/32  S   5                       >192.168.71.254

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route no-community

Syntax
show route no-community
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route no-community
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the route entries in each routing table that are not associated with any community.

Options
none—(Same as **brief**) Display the route entries in each routing table that are not associated with any community.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route no-community on page 605
show route no-community detail on page 606
show route no-community extensive on page 606
show route no-community terse on page 607

Output Fields
For information about output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route no-community

```
user@host> show route no-community
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.0.0/16       *[Static/5] 00:36:27
                    > to 192.168.71.254 via fxp0.0
10.209.0.0/16      *[Static/5] 00:36:27
                    > to 192.168.71.254 via fxp0.0
10.255.71.52/32    *[Direct/0] 00:36:27
                    > via lo0.0
10.255.71.63/32    *[OSPF/10] 00:04:39, metric 1
                    > to 35.1.1.2 via ge-3/1/0.0
10.255.71.64/32    *[OSPF/10] 00:00:08, metric 2
                    > to 35.1.1.2 via ge-3/1/0.0
10.255.71.240/32   *[OSPF/10] 00:05:04, metric 2
                       via so-0/1/2.0
```

```
                    > via so-0/3/2.0
10.255.71.241/32    *[OSPF/10] 00:05:14, metric 1
                    > via so-0/1/2.0
10.255.71.242/32    *[OSPF/10] 00:05:19, metric 1
                    > via so-0/3/2.0
12.1.1.0/24         *[OSPF/10] 00:05:14, metric 2
                    > via so-0/3/2.0
14.1.1.0/24         *[OSPF/10] 00:00:08, metric 3
                    > to 35.1.1.2 via ge-3/1/0.0
                      via so-0/1/2.0
                      via so-0/3/2.0
16.1.1.0/24         *[OSPF/10] 00:05:14, metric 2
                    > via so-0/1/2.0
.....
```

show route no-community detail

```
user@host> show route no-community detail

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Age: 38:08
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Age: 38:08
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
....
```

show route no-community extensive

```
user@host> show route no-community extensive

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.10.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:03:33
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
TSI:
```

```
KRT in-kernel 10.209.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:03:33
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
```

show route no-community terse

```
user@host> show route no-community terse

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.10.0.0/16       S   5                       >192.168.71.254
* 10.209.0.0/16      S   5                       >192.168.71.254
* 10.255.71.52/32    D   0                       >lo0.0
* 10.255.71.63/32    O  10          1            >35.1.1.2
* 10.255.71.64/32    O  10          2            >35.1.1.2
* 10.255.71.240/32   O  10          2             so-0/1/2.0
                                                 >so-0/3/2.0
* 10.255.71.241/32   O  10          1            >so-0/1/2.0
* 10.255.71.242/32   O  10          1            >so-0/3/2.0
* 12.1.1.0/24        O  10          2            >so-0/3/2.0
* 14.1.1.0/24        O  10          3            >35.1.1.2
                                                  so-0/1/2.0
                                                  so-0/3/2.0
* 16.1.1.0/24        O  10          2            >so-0/1/2.0
...
```

show route output

Syntax

show route output (address *ip-address* | interface *interface-name*)
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route output (address *ip-address* | interface *interface-name*)
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the entries in the routing table learned through static routes and interior gateway protocols that are to be sent out the interface with either the specified IP address or specified name.

To view routes advertised to a neighbor or received from a neighbor for the BGP protocol, use the **show route advertising-protocol bgp** and **show route receive-protocol bgp** commands instead.

Options

address *ip-address*—Display entries in the routing table that are to be sent out the interface with the specified IP address.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

interface *interface-name*—Display entries in the routing table that are to be sent out the interface with the specified name.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route output address on page 609
show route output address detail on page 609
show route output address extensive on page 610
show route output address terse on page 610
show route output interface on page 610
show route output interface detail on page 611
show route output interface extensive on page 611
show route output interface terse on page 611

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route output address

```
user@host> show route output address 36.1.1.1/24

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

36.1.1.0/24        *[Direct/0] 00:19:56
                    > via so-0/1/2.0
                    [OSPF/10] 00:19:55, metric 1
                    > via so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 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)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output address detail

```
user@host> show route output address 36.1.1.1 detail

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
36.1.1.0/24 (2 entries, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via so-0/1/2.0, selected
                State: <Active Int>
                Age: 23:00
                Task: IF
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/1/2.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Age: 22:59      Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 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)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output address extensive

The output for the **show route output address extensive** command is identical to that of the **show route output address detail** command. For sample output, see show route output address detail on page 609.

show route output address terse

```
user@host> show route output address 36.1.1.1 terse

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 36.1.1.0/24        D   0                     >so-0/1/2.0
                     O  10          1          >so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 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)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output interface

```
user@host> show route output interface so-0/1/2.0

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.71.240/32   *[OSPF/10] 00:13:00, metric 2
                      via so-0/1/2.0
                    > via so-0/3/2.0
10.255.71.241/32   *[OSPF/10] 00:13:10, metric 1
                    > via so-0/1/2.0
14.1.1.0/24        *[OSPF/10] 00:05:11, metric 3
                      to 35.1.1.2 via ge-3/1/0.0
                    > via so-0/1/2.0
                      via so-0/3/2.0
16.1.1.0/24        *[OSPF/10] 00:13:10, metric 2
                    > via so-0/1/2.0
36.1.1.0/24        *[Direct/0] 00:13:21
                    > via so-0/1/2.0
                     [OSPF/10] 00:13:20, metric 1
                    > via so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 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)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output interface detail

```
user@host> show route output interface so-0/1/2.0 detail

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
10.255.71.240/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 2
                Next hop: via so-0/1/2.0
                Next hop: via so-0/3/2.0, selected
                State: <Active Int>
                Age: 14:52      Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I

10.255.71.241/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 4
                Next hop: via so-0/1/2.0, selected
                State: <Active Int>
                Age: 15:02      Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I
...
```

show route output interface extensive

The output for the **show route output interface extensive** command is identical to that of the **show route output interface detail** command. For sample output, see show route output interface detail on page 611.

show route output interface terse

```
user@host> show route output interface so-0/1/2.0 terse

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.71.240/32   O  10          2             so-0/1/2.0
                                                 >so-0/3/2.0
* 10.255.71.241/32   O  10          1            >so-0/1/2.0
* 14.1.1.0/24        O  10          3             35.1.1.2
                                                 >so-0/1/2.0
                                                  so-0/3/2.0
* 16.1.1.0/24        O  10          2            >so-0/1/2.0
* 36.1.1.0/24        D   0                       >so-0/1/2.0
                     O  10          1            >so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 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)
```

```
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route protocol

Syntax

show route protocol *protocol*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route protocol *protocol*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
ospf2 and **ospf3** options introduced in Junos OS Release 9.2.
ospf2 and **ospf3** options introduced in Junos OS Release 9.2 for EX Series switches.
flow option introduced in Junos OS Release 10.0.
flow option introduced in Junos OS Release 10.0 for EX Series switches.

Description

Display the route entries in the routing table that were learned from a particular protocol.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

protocol—Protocol from which the route was learned:

- **access**—Access route for use by DHCP application
- **access-internal**—Access-internal route for use by DHCP application
- **aggregate**—Locally generated aggregate route
- **arp**—Route learned through the Address Resolution Protocol
- **atmvpn**—Asynchronous Transfer Mode virtual private network
- **bgp**—Border Gateway Protocol
- **ccc**—Circuit cross-connect
- **direct**—Directly connected route
- **dvmrp**—Distance Vector Multicast Routing Protocol
- **esis**—End System-to-Intermediate System
- **flow**—Locally defined flow-specification route
- **frr**—Precomputed protection route or backup route used when a link goes down
- **isis**—Intermediate System-to-Intermediate System
- **ldp**—Label Distribution Protocol
- **l2circuit**—Layer 2 circuit
- **l2vpn**—Layer 2 virtual private network

- **local**—Local address
- **mpls**—Multiprotocol Label Switching
- **msdp**—Multicast Source Discovery Protocol
- **ospf**—Open Shortest Path First versions 2 and 3
- **ospf2**—Open Shortest Path First versions 2 only
- **ospf3**—Open Shortest Path First version 3 only
- **pim**—Protocol Independent Multicast
- **rip**—Routing Information Protocol
- **ripng**—Routing Information Protocol next generation
- **rsvp**—Resource Reservation Protocol
- **rtarget**—Local route target virtual private network
- **static**—Statically defined route
- **tunnel**—Dynamic tunnel
- **vpn**—Virtual private network

NOTE: EX Series switches run a subset of these protocols. See the switch CLI for details.

Required Privilege Level view

List of Sample Output

show route protocol access on page 615
show route protocol access-internal extensive on page 615
show route protocol arp on page 615
show route protocol bgp on page 616
show route protocol bgp detail on page 616
show route protocol bgp extensive on page 616
show route protocol bgp terse on page 617
show route protocol direct on page 617
show route protocol frr on page 618
show route protocol l2circuit detail on page 618
show route protocol l2vpn extensive on page 619
show route protocol ldp on page 620
show route protocol ldp extensive on page 620
show route protocol ospf (Layer 3 VPN) on page 621
show route protocol ospf detail on page 622
show route protocol rip on page 622
show route protocol rip detail on page 622
show route protocol ripng table inet6 on page 623
show route protocol static detail on page 623

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route protocol access

```
user@host> show route protocol access
inet.0: 30380 destinations, 30382 routes (30379 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

13.160.0.3/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
13.160.0.4/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
13.160.0.5/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
```

show route protocol access-internal extensive

```
user@host> show route protocol access-internal 13.160.0.19 extensive
inet.0: 100020 destinations, 100022 routes (100019 active, 0 holddown, 1 hidden)
13.160.0.19/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 13.160.0.19/32 -> {13.160.0.2}
        *Access-internal Preference: 12
                Next-hop reference count: 200000
                Next hop: 13.160.0.2 via fe-0/0/0.0, selected
                State: <Active Int>
       Age: 36
                Task: RPD Unix Domain Server./var/run/rpd_serv.local
                Announcement bits (1): 0-KRT
                AS path: I
```

show route protocol arp

```
user@host> show route protocol arp
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

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

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.20.1.3/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.4/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.5/32        [ARP/4294967293] 00:04:32, from 20.20.1.1
                      Unusable
20.20.1.6/32        [ARP/4294967293] 00:04:34, from 20.20.1.1
                      Unusable
20.20.1.7/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.8/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.9/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.10/32       [ARP/4294967293] 00:04:35, from 20.20.1.1
```

```
                      Unusable
20.20.1.11/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
20.20.1.12/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
20.20.1.13/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
...
```

show route protocol bgp

```
user@host> show route protocol bgp 192.168.64.0/21
inet.0: 335832 destinations, 335833 routes (335383 active, 0 holddown, 450 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.64.0/21       *[BGP/170] 6d 10:41:16, localpref 100, from 192.168.69.71
                      AS path: 10458 14203 2914 4788 4788 I
                    > to 192.168.167.254 via fxp0.0
```

show route protocol bgp detail

```
user@host> show route protocol bgp 66.117.63.0/24 detail
inet.0: 335805 destinations, 335806 routes (335356 active, 0 holddown, 450 hidden)
66.117.63.0/24  (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 1006436
                Source: 192.168.69.71
                Next hop type: Router, Next hop index: 324
                Next hop: 192.168.167.254 via fxp0.0, selected
                Protocol next hop: 192.168.69.71
                Indirect next hop: 8e166c0 342
                State: <Active Ext>
                Local AS:    69 Peer AS: 10458
                Age: 6d 10:42:42        Metric2: 0
                Task: BGP_10458.192.168.69.71+179
                Announcement bits (3): 0-KRT 2-BGP RT Background 3-Resolve tree
1
                AS path: 10458 14203 2914 4788 4788 I
                Communities: 2914:410 2914:2403 2914:3400
                Accepted
                Localpref: 100
                Router ID: 207.17.136.192
```

show route protocol bgp extensive

```
user@host> show route protocol bgp 192.168.64.0/21 extensive

inet.0: 335827 destinations, 335828 routes (335378 active, 0 holddown, 450 hidden)
192.168.64.0/21 (1 entry, 1 announced)
TSI:
KRT in-kernel 1.9.0.0/16 -> {indirect(342)}
Page 0 idx 1 Type 1 val db31a80
    Nexthop: Self
    AS path: [69] 10458 14203 2914 4788 4788 I
    Communities: 2914:410 2914:2403 2914:3400
Path 1.9.0.0 from 192.168.69.71 Vector len 4.  Val: 1
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 1006502
                Source: 192.168.69.71
                Next hop type: Router, Next hop index: 324
```

```
                    Next hop: 192.168.167.254 via fxp0.0, selected
                    Protocol next hop: 192.168.69.71
                    Indirect next hop: 8e166c0 342
                    State: <Active Ext>
                    Local AS:    69 Peer AS: 10458
                    Age: 6d 10:44:45        Metric2: 0
                    Task: BGP_10458.192.168.69.71+179
                    Announcement bits (3): 0-KRT 2-BGP RT Background 3-Resolve tree
1
                    AS path: 10458 14203 2914 4788 4788 I
                    Communities: 2914:410 2914:2403 2914:3400
                    Accepted
                    Localpref: 100
                    Router ID: 207.17.136.192
                    Indirect next hops: 1
                            Protocol next hop: 192.168.69.71
                            Indirect next hop: 8e166c0 342
                            Indirect path forwarding next hops: 1
                                    Next hop type: Router
                                    Next hop: 192.168.167.254 via fxp0.0
                            192.168.0.0/16 Originating RIB: inet.0
                              Node path count: 1
                              Forwarding nexthops: 1
                                    Nexthop: 192.168.167.254 via fxp0.0
```

show route protocol bgp terse

```
user@host> show route protocol bgp 192.168.64.0/21 terse

inet.0: 24 destinations, 32 routes (23 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  192.168.64.0/21    B 170        100            >100.1.3.2       10023 21 I
```

show route protocol direct

```
user@host> show route protocol direct

inet.0: 335843 destinations, 335844 routes (335394 active, 0 holddown, 450 hidden)
+ = Active Route, - = Last Active, * = Both

8.8.8.0/24         *[Direct/0] 17w0d 10:31:49
                    > via fe-1/3/1.0
10.255.165.1/32    *[Direct/0] 25w4d 04:13:18
                    > via lo0.0
30.30.30.0/24      *[Direct/0] 17w0d 23:06:26
                    > via fe-1/3/2.0
192.168.164.0/22   *[Direct/0] 25w4d 04:13:20
                    > via fxp0.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

47.0005.80ff.f800.0000.0108.0001.0102.5516.5001/152
                   *[Direct/0] 25w4d 04:13:21
                    > via lo0.0

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
abcd::10:255:165:1/128
                   *[Direct/0] 25w4d 04:13:21
                    > via lo0.0
fe80::2a0:a5ff:fe12:ad7/128
                   *[Direct/0] 25w4d 04:13:21
                    > via lo0.0
```

show route protocol frr

```
user@host> show route protocol frr
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

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

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.20.1.3/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.3 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.4/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.4 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.5/32       *[FRR/200] 00:05:35, from 20.20.1.1
                    > to 20.20.1.5 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.6/32       *[FRR/200] 00:05:37, from 20.20.1.1
                    > to 20.20.1.6 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.7/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.7 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.8/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.8 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.9/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.9 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.10/32      *[FRR/200] 00:05:38, from 20.20.1.1
...
```

show route protocol l2circuit detail

```
user@host> show route protocol l2circuit detail

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
100000 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via ge-2/0/0.0, selected
                Label operation: Pop       Offset: 4
                State: <Active Int>
                Local AS:    99
                Age: 9:52
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

ge-2/0/0.0 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
```

```
                    Label operation: Push 100000, Push 100000(top)[0] Offset: -4
                    Protocol next hop: 10.245.255.63
                    Push 100000 Offset: -4
                     Indirect next hop: 86af0c0 298
                    State: <Active Int>
                    Local AS:    99
                    Age: 9:52
                    Task: Common L2 VC
                    Announcement bits (2): 0-KRT 1-Common L2 VC
                    AS path: I

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

10.245.255.63:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
                Protocol next hop: 10.245.255.63 Indirect next hop: 86af000 296
                State: <Active Int>
                Local AS:    99
                Age: 10:21
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 100000, MTU 1500, VLAN ID 512
```

show route protocol l2vpn extensive

```
user@host> show route protocol l2vpn extensive

inet.0: 14 destinations, 15 routes (13 active, 0 holddown, 1 hidden)

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: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
800001 (1 entry, 1 announced)
TSI:
KRT in-kernel 800001 /36 -> {so-0/0/0.0}
        *L2VPN  Preference: 7
                Next hop: via so-0/0/0.0 weight 49087 balance 97%, selected
                Label operation: Pop       Offset: 4
                State: <Active Int>
                Local AS:    69
                Age: 7:48
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

so-0/0/0.0 (1 entry, 1 announced)
TSI:
KRT in-kernel so-0/0/0.0.0      /16 -> {indirect(288)}
        *L2VPN  Preference: 7
                Next hop: via so-0/0/1.0, selected
                Label operation: Push 800000 Offset: -4
                Protocol next hop: 10.255.14.220
                Push 800000 Offset: -4
                 Indirect next hop: 85142a0 288
                State: <Active Int>
```

```
                    Local AS:    69
                    Age: 7:48
                    Task: Common L2 VC
                    Announcement bits (2): 0-KRT 1-Common L2 VC
                    AS path: I
                    Communities: target:69:1 Layer2-info: encaps:PPP,
                    control flags:2, mtu: 0
```

show route protocol ldp

```
user@host> show route protocol ldp
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.16.1/32    *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Push 100000
192.168.17.1/32    *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0

private1__.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100064             *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Pop
100064(S=0)        *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Pop
100080             *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Swap 100000
```

show route protocol ldp extensive

```
user@host> show route protocol ldp extensive
192.168.16.1/32 (1 entry, 1 announced)
        State: <FlashAll>
        *LDP    Preference: 9
                Next-hop reference count: 3
                Next hop: via t1-4/0/0.0, selected
                Label operation: Push 100000
                State: <Active Int>
                Local AS: 65500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (2): 0-Resolve tree 1 2-Resolve tree 2
                AS path: I

192.168.17.1/32 (1 entry, 1 announced)
        State: <FlashAll>
        *LDP    Preference: 9
                Next-hop reference count: 3
                Next hop: via t1-4/0/0.0, selected
                State: <Active Int>
                Local AS: 65500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (2): 0-Resolve tree 1 2-Resolve tree 2
                AS path: I
```

```
private1__.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

100064 (1 entry, 1 announced)
TSI:
KRT in-kernel 100064 /36 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                State: <Active Int>
                Local AS: 65500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 192.168.17.1/32

100064(S=0) (1 entry, 1 announced)
TSI:
KRT in-kernel 100064 /40 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                Label operation: Pop
                State: <Active Int>
                Local AS: 65500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I

100080 (1 entry, 1 announced)
TSI:
KRT in-kernel 100080 /36 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                Label operation: Swap 100000
                State: <Active Int>
                Local AS: 65500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 192.168.16.1/32
```

show route protocol ospf (Layer 3 VPN)

```
user@host> show route protocol ospf
inet.0: 40 destinations, 40 routes (39 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.4/30       *[OSPF/10] 00:05:18, metric 4
                    > via t3-3/2/0.0
10.39.1.8/30        [OSPF/10] 00:05:18, metric 2
                    > via t3-3/2/0.0
10.255.14.171/32   *[OSPF/10] 00:05:18, metric 4
                    > via t3-3/2/0.0
10.255.14.179/32   *[OSPF/10] 00:05:18, metric 2
                    > via t3-3/2/0.0
```

```
224.0.0.5/32       *[OSPF/10] 20:25:55, metric 1

VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.16/30       [OSPF/10] 00:05:43, metric 1
                    > via so-0/2/2.0
10.255.14.173/32   *[OSPF/10] 00:05:43, metric 1
                    > via so-0/2/2.0
224.0.0.5/32       *[OSPF/10] 20:26:20, metric 1
```

show route protocol ospf detail

```
user@host> show route protocol ospf detail
VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.16/30 (2 entries, 0 announced)
         OSPF   Preference:  10
                Nexthop: via so-0/2/2.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Age: 6:25       Metric: 1
                Area: 0.0.0.0
                Task: VPN-AB-OSPF
                AS path: I
                Communities: Route-Type:0.0.0.0:1:0

...
```

show route protocol rip

```
user@host> show route protocol rip
inet.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.255.14.177/32   *[RIP/100] 20:24:34, metric 2
                    > to 10.39.1.22 via t3-0/2/2.0
224.0.0.9/32       *[RIP/100] 00:03:59, metric 1
```

show route protocol rip detail

```
user@host> show route protocol rip detail
inet.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.255.14.177/32 (1 entry, 1 announced)
        *RIP    Preference: 100
                Nexthop: 10.39.1.22 via t3-0/2/2.0, selected
                State: <Active Int>
                Age: 20:25:02   Metric: 2
                Task: VPN-AB-RIPv2
                Announcement bits (2): 0-KRT 2-BGP.0.0.0.0+179
                AS path: I
                Route learned from 10.39.1.22 expires in 96 seconds
```

show route protocol ripng table inet6

```
user@host> show route protocol ripng table inet6
inet6.0: 4215 destinations, 4215 routes (4214 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

1111::1/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::2/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::3/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::4/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::5/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::6/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
```

show route protocol static detail

```
user@host> show route protocol static detail
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
10.5.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.13.10.0/23 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
                Validation State: unverified
```

```
Task: RT
Announcement bits (1): 0-KRT
AS path: I
```

show route receive-protocol

Syntax
show route receive-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)

Syntax (EX Series Switches)
show route receive-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the routing information as it was received through a particular neighbor using a particular dynamic routing protocol.

Options
brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

protocol neighbor-address—Protocol transmitting the route (**bgp**, **dvmrp**, **msdp**, **pim**, **rip**, or **ripng**) and address of the neighboring router from which the route entry was received.

Additional Information
The output displays the selected routes and the attributes with which they were received, but does not show the effects of import policy on the routing attributes.

Required Privilege Level
view

List of Sample Output
show route receive-protocol bgp on page 628
show route receive-protocol bgp extensive on page 628
show route receive-protocol bgp table extensive on page 628
show route receive-protocol bgp logical-system extensive on page 629
show route receive-protocol bgp detail (Layer 2 VPN) on page 630
show route receive-protocol bgp extensive (Layer 2 VPN) on page 630
show route receive-protocol bgp (Layer 3 VPN) on page 631
show route receive-protocol bgp detail (Layer 3 VPN) on page 631
show route receive-protocol bgp extensive (Layer 3 VPN) on page 632

Output Fields
Table 41 on page 625 describes the output fields for the **show route receive-protocol** command. Output fields are listed in the approximate order in which they appear.

Table 41: show route receive-protocol Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, inet.0.	All levels
number destinations	Number of destinations for which there are routes in the routing table.	All levels

Table 41: show route receive-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** • **holddown** (routes that are in pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)	All levels
Prefix	Destination prefix.	none **brief**
MED	Multiple exit discriminator value included in the route.	none **brief**
destination-prefix (entry, announced)	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.	**detail extensive**
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.	**detail extensive**
VPN Label	Virtual private network (VPN) label. Packets are sent between CE and PE routing devices by advertising VPN labels. VPN labels transit over either an RSVP or an LDP label-switched path (LSP) tunnel.	**detail extensive**
Next hop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.	All levels
Localpref or Lclpref	Local preference value included in the route.	All levels

Table 41: show route receive-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
AS path	Autonomous system (AS) path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the router, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
Cluster list	(For route reflected output only) Cluster ID sent by the route reflector.	**detail extensive**
Originator ID	(For route reflected output only) Address of routing device that originally sent the route to the route reflector.	**detail extensive**
Communities	Community path attribute for the route. See the Output Field table in the show route detail command for all possible values for this field.	**detail extensive**
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.	**detail extensive**
Attrset AS	Number, local preference, and path of the AS that originated the route. These values are stored in the **Attrset** attribute at the originating routing device.	**detail extensive**
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).	**detail extensive**
control flags	Control flags: **none** or **Site Down**.	**detail extensive**
mtu	Maximum transmission unit (MTU) of the Layer 2 circuit.	**detail extensive**

Sample Output

show route receive-protocol bgp

```
user@host> show route receive-protocol bgp 10.255.245.215

inet.0: 28 destinations, 33 routes (27 active, 0 holddown, 1 hidden)
Prefix                  Next hop              MED     Lclpref    AS path
10.22.1.0/24            10.255.245.215        0       100        I
10.22.2.0/24            10.255.245.215        0       100        I
```

show route receive-protocol bgp extensive

```
user@host> show route receive-protocol bgp 10.255.245.63 extensive
inet.0: 244 destinations, 244 routes (243 active, 0 holddown, 1 hidden)
Prefix             Next hop               MED    Lclpref AS path
1.1.1.0/24 (1 entry, 1 announced)
     Next hop: 10.0.50.3
     Localpref: 100
     AS path: I <Orginator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.45
165.3.0.0/16 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.68
165.4.0.0/16 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.45
195.1.2.0/24 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.68
inet.2: 63 destinations, 63 routes (63 active, 0 holddown, 0 hidden)
Prefix             Next hop               MED    Lclpref AS path
inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Next hop               MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Next hop               MED    Lclpref AS path
mpls.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
```

show route receive-protocol bgp table extensive

```
user@host> show route receive-protocol bgp 207.17.136.192 table inet.0 66.117.68.0/24 extensive
inet.0: 227315 destinations, 227316 routes (227302 active, 0 holddown, 13 hidden)
* 66.117.63.0/24 (1 entry, 1 announced)
     Nexthop: 207.17.136.29
     Localpref: 100
     AS path: AS2 PA[6]: 14203 2914 3356 29748 33437 AS_TRANS
     AS path: AS4 PA[2]: 33437 393219
     AS path: Merged[6]: 14203 2914 3356 29748 33437 393219 I
     Communities: 2914:420
```

show route receive-protocol bgp logical-system extensive

```
user@host> show route receive-protocol bgp 10.0.0.9 logical-system PE4 extensive
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
* 10.0.0.0/30 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.0.0.4/30 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

  10.0.0.8/30 (2 entries, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.9.9.1/32 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.100.1.1/32 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 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

* 66.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300144
     Nexthop: 10.0.0.9
     AS path: 13979 7018 I

* 99.0.0.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300160
     Nexthop: 10.0.0.9
     AS path: 13979 7018 I
```

show route receive-protocol bgp detail (Layer 2 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171 detail
inet.0: 68 destinations, 68 routes (67 active, 0 holddown, 1 hidden)
Prefix             Nexthop               MED    Lclpref AS path
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
frame-vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
Prefix             Nexthop               MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 1 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags: 0, mtu: 0
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 0 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
```

show route receive-protocol bgp extensive (Layer 2 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171 extensive
inet.0: 68 destinations, 68 routes (67 active, 0 holddown, 1 hidden)
Prefix             Nexthop               MED    Lclpref AS path
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
frame-vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 1 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop               MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 0 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
```

```
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
```

show route receive-protocol bgp (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171
      inet.0: 33 destinations, 33 routes (32 active, 0 holddown, 1 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      VPN-A.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      10.255.14.175/32   10.255.14.171                    100 2 I
      10.255.14.179/32   10.255.14.171           2        100 I
      VPN-B.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      10.255.14.175/32   10.255.14.171                    100 2 I
      10.255.14.177/32   10.255.14.171                    100 I
      iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      mpls.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      bgp.l3vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
      Prefix             Nexthop               MED    Lclpref AS path
      10.255.14.171:300:10.255.14.177/32
                         10.255.14.171                    100 I
      10.255.14.171:100:10.255.14.179/32
                         10.255.14.171           2        100 I
      10.255.14.171:200:10.255.14.175/32
                         10.255.14.171                    100 2 I
```

show route receive-protocol bgp detail (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.174 detail
inet.0: 16 destinations, 17 routes (15 active, 0 holddown, 1 hidden)
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
vpna.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 10.49.0.0/30 (1 entry, 1 announced)
     Route Distinguisher: 10.255.14.176:2
     VPN Label: 101264
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
* 10.255.14.172/32 (1 entry, 1 announced)
     Route Distinguisher: 10.255.14.176:2
     VPN Label: 101280
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
* 10.255.14.174:2:10.49.0.0/30 (1 entry, 0 announced)
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101264
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
* 10.255.14.174:2:10.255.14.172/32 (1 entry, 0 announced)
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101280
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

show route receive-protocol bgp extensive (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.245.63 extensive
inet.0: 244 destinations, 244 routes (243 active, 0 holddown, 1 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     1.1.1.0/24 (1 entry, 1 announced)
          Nexthop: 10.0.50.3
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.45
     165.3.0.0/16 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.68
     165.4.0.0/16 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.45
     195.1.2.0/24 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.68
     inet.2: 63 destinations, 63 routes (63 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     mpls.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
```

show route table

Syntax

show route table *routing-table-name*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route table *routing-table-name*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the route entries in a particular routing table.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

routing-table-name—Display route entries for all routing tables whose name begins with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route table inet** command).

Required Privilege Level

view

Related Documentation

- *show route summary*

List of Sample Output

show route table bgp.l2.vpn on page 634
show route table bgp.l3vpn.0 on page 634
show route table bgp.l3vpn.0 detail on page 634
show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured) on page 636
show route table inet.0 on page 636
show route table inet6.0 on page 636
show route table inet6.3 on page 637
show route table inetflow detail on page 637
show route table l2circuit.0 on page 637
show route table mpls on page 638
show route table mpls extensive on page 638
show route table mpls.0 on page 638
show route table mpls.0 (RSVP Route—Transit LSP) on page 639
show route table vpls_1 detail on page 639
show route table vpn-a on page 639
show route table vpn-a.mdt.0 on page 640
show route table VPN-A detail on page 640
show route table VPN-AB.inet.0 on page 640
show route table VPN_blue.mvpn-inet6.0 on page 641
show route table VPN-A detail on page 641

show route table inetflow detail on page 642

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route table bgp.l2.vpn

```
user@host> show route table bgp.l2.vpn
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.24.1:1:4:1/96
                   *[BGP/170] 01:08:58, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table bgp.l3vpn.0

```
user@host> show route table bgp.l3vpn.0
      bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
      + = Active Route, - = Last Active, * = Both

      10.255.71.15:100:10.255.71.17/32
                         *[BGP/170] 00:03:59, MED 1, localpref 100, from
      10.255.71.15
                            AS path: I
                          > via so-2/1/0.0, Push 100020, Push 100011(top)
      10.255.71.15:200:10.255.71.18/32
                         *[BGP/170] 00:03:59, MED 1, localpref 100, from
      10.255.71.15
                            AS path: I
                          > via so-2/1/0.0, Push 100021, Push 100011(top)
```

show route table bgp.l3vpn.0 detail

```
user@host> show route table bgp.l3vpn.0 detail
bgp.l3vpn.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)

10.255.245.12:1:4.0.0.0/8 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
         Route Distinguisher: 10.255.245.12:1
         Source: 10.255.245.12
         Next hop: 192.168.208.66 via fe-0/0/0.0, selected
         Label operation: Push 182449
         Protocol next hop: 10.255.245.12
         Push 182449
         Indirect next hop: 863a630 297
         State: <Active Int Ext>
         Local AS:    35 Peer AS:    35
         Age: 12:19      Metric2: 1
         Task: BGP_35.10.255.245.12+179
         Announcement bits (1): 0-BGP.0.0.0.0+179
         AS path: 30 10458 14203 2914 3356 I (Atomic) Aggregator: 3356 4.68.0.11

         Communities: 2914:420 target:11111:1 origin:56:78
         VPN Label: 182449
         Localpref: 100
```

```
          Router ID: 10.255.245.12

10.255.245.12:1:4.17.225.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
         Route Distinguisher: 10.255.245.12:1
         Source: 10.255.245.12
         Next hop: 192.168.208.66 via fe-0/0/0.0, selected
         Label operation: Push 182465
         Protocol next hop: 10.255.245.12
         Push 182465
         Indirect next hop: 863a8f0 305
         State: <Active Int Ext>
         Local AS:    35 Peer AS:    35
         Age: 12:19      Metric2: 1
         Task: BGP_35.10.255.245.12+179
         Announcement bits (1): 0-BGP.0.0.0.0+179
AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496 6496 I
                Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
                VPN Label: 182465
                Localpref: 100
                Router ID: 10.255.245.12

10.255.245.12:1:4.17.226.0/23 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
         Route Distinguisher: 10.255.245.12:1
         Source: 10.255.245.12
         Next hop: 192.168.208.66 via fe-0/0/0.0, selected
         Label operation: Push 182465
         Protocol next hop: 10.255.245.12
         Push 182465
         Indirect next hop: 86bd210 330
         State: <Active Int Ext>
         Local AS:    35 Peer AS:    35
         Age: 12:19      Metric2: 1
         Task: BGP_35.10.255.245.12+179
         Announcement bits (1): 0-BGP.0.0.0.0+179
         AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

         6496 I
         Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
         VPN Label: 182465
         Localpref: 100
         Router ID: 10.255.245.12

10.255.245.12:1:4.17.251.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
         Route Distinguisher: 10.255.245.12:1
         Source: 10.255.245.12
         Next hop: 192.168.208.66 via fe-0/0/0.0, selected
         Label operation: Push 182465
         Protocol next hop: 10.255.245.12
         Push 182465
         Indirect next hop: 86bd210 330
         State: <Active Int Ext>
         Local AS:    35 Peer AS:    35
         Age: 12:19      Metric2: 1
         Task: BGP_35.10.255.245.12+179
         Announcement bits (1): 0-BGP.0.0.0.0+179
         AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

         6496 I
```

```
                    Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
                    VPN Label: 182465
                    Localpref: 100
```

show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured)

```
user@host> show route table bgp.rtarget.o
bgp.rtarget.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100:100:100/96
                   *[RTarget/5] 00:03:14
                      Type Proxy
                        for 10.255.165.103
                        for 10.255.166.124
                      Local
```

show route table inet.0

```
user@host> show route table inet.0
inet.0: 12 destinations, 12 routes (11 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:51:57
                    > to 111.222.5.254 via fxp0.0
1.0.0.1/32         *[Direct/0] 00:51:58
                    > via at-5/3/0.0
1.0.0.2/32         *[Local/0] 00:51:58
                      Local
12.12.12.21/32     *[Local/0] 00:51:57
                      Reject
13.13.13.13/32     *[Direct/0] 00:51:58
                    > via t3-5/2/1.0
13.13.13.14/32     *[Local/0] 00:51:58
                      Local
13.13.13.21/32     *[Local/0] 00:51:58
                      Local
13.13.13.22/32     *[Direct/0] 00:33:59
                    > via t3-5/2/0.0
127.0.0.1/32        [Direct/0] 00:51:58
                    > via lo0.0
111.222.5.0/24     *[Direct/0] 00:51:58
                    > via fxp0.0
111.222.5.81/32    *[Local/0] 00:51:58
                      Local
```

show route table inet6.0

```
user@host> show route table inet6.0
inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Route, * = Both

fec0:0:0:3::/64 *[Direct/0] 00:01:34
>via fe-0/1/0.0

fec0:0:0:3::/128 *[Local/0] 00:01:34
>Local

fec0:0:0:4::/64 *[Static/5] 00:01:34
>to fec0:0:0:3::ffff via fe-0/1/0.0
```

show route table inet6.3

```
user@router> show route table inet6.3
inet6.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

::10.255.245.195/128
                   *[LDP/9] 00:00:22, metric 1
                    > via so-1/0/0.0
::10.255.245.196/128
                   *[LDP/9] 00:00:08, metric 1
                    > via so-1/0/0.0, Push 100008
```

show route table inetflow detail

```
user@host> show route table inetflow detail
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 2
                State: <Active Ext>
                Local AS: 65002 Peer AS: 65000
                Age: 4
                Task: BGP_65000.10.12.99.5+3792
                Announcement bits (1): 0-Flow
                AS path: 65000 I
                Communities: traffic-rate:0:0
                Validation state: Accept, Originator: 10.12.99.5
                Via: 10.12.44.0/24, Active
                Localpref: 100
                Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
        *Flow   Preference: 5
                Next-hop reference count: 2
                State: <Active>
                Local AS: 65002
                Age: 6:30
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
                AS path: I
                Communities: 1:1
```

show route table l2circuit.0

```
user@host> show route table l2circuit.0
l2circuit.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.195:NoCtrlWord:1:1:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:NoCtrlWord:1:1:Remote/96
                   *[LDP/9] 00:50:14
                      Discard
10.1.1.195:CtrlWord:1:2:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:CtrlWord:1:2:Remote/96
```

```
                        *[LDP/9] 00:50:14
                           Discard
```

show route table mpls

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

0                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1                  *[MPLS/0] 00:13:55, metric 1
                      Receive
2                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1024               *[VPN/0] 00:04:18
                      to table red.inet.0, Pop
```

show route table mpls extensive

```
user@host> show route table mpls extensive
100000 (1 entry, 1 announced)
TSI:
KRT in-kernel 100000 /36 -> {so-1/0/0.0}
        *LDP    Preference:   9
                Next hop:  via so-1/0/0.0, selected
                Pop
                State: <Active Int>
                Age: 29:50      Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 10.0.0.194/32
```

show route table mpls.0

```
user@host> show route table mpls.0
mpls.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:45:09, metric 1
                      Receive
1                  *[MPLS/0] 00:45:09, metric 1
                      Receive
2                  *[MPLS/0] 00:45:09, metric 1
                      Receive
100000             *[L2VPN/7] 00:43:04
                    > via so-0/1/0.1, Pop
100001             *[L2VPN/7] 00:43:03
                    > via so-0/1/0.2, Pop       Offset: 4
100002             *[LDP/9] 00:43:22, metric 1
                      via so-0/1/2.0, Pop
                    > via so-0/1/3.0, Pop
100002(S=0)        *[LDP/9] 00:43:22, metric 1
                      via so-0/1/2.0, Pop
                    > via so-0/1/3.0, Pop
100003             *[LDP/9] 00:43:22, metric 1
                    > via so-0/1/2.0, Swap 100002
                      via so-0/1/3.0, Swap 100002
100004             *[LDP/9] 00:43:16, metric 1
                      via so-0/1/2.0, Swap 100049
                    > via so-0/1/3.0, Swap 100049
```

```
so-0/1/0.1          *[L2VPN/7] 00:43:04
                     > via so-0/1/2.0, Push 100001, Push 100049(top)
                       via so-0/1/3.0, Push 100001, Push 100049(top)
so-0/1/0.2          *[L2VPN/7] 00:43:03
                       via so-0/1/2.0, Push 100000, Push 100049(top) Offset: -4
                     > via so-0/1/3.0, Push 100000, Push 100049(top) Offset: -4
```

show route table mpls.0 (RSVP Route—Transit LSP)

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

mpls.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                   *[MPLS/0] 00:37:31, metric 1
                       Receive
1                   *[MPLS/0] 00:37:31, metric 1
                       Receive
2                   *[MPLS/0] 00:37:31, metric 1
                       Receive
13                  *[MPLS/0] 00:37:31, metric 1
                       Receive
300352              *[RSVP/7/1] 00:08:00, metric 1
                     > to 8.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300352(S=0)         *[RSVP/7/1] 00:08:00, metric 1
                     > to 8.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300384              *[RSVP/7/2] 00:05:20, metric 1
                     > to 8.64.1.106 via ge-1/0/0.0, Pop
300384(S=0)         *[RSVP/7/2] 00:05:20, metric 1
                     > to 8.64.1.106 via ge-1/0/0.0, Pop
```

show route table vpls_1 detail

```
user@host> show route table vpls_1 detail
vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

1.1.1.11:1000:1:1/96 (1 entry, 1 announced)
*L2VPN Preference: 170/-1
Receive table: vpls_1.l2vpn.0
Next-hop reference count: 2
State: <Active Int Ext>
Age: 4:29:47 Metric2: 1
Task: vpls_1-l2vpn
Announcement bits (1): 1-BGP.0.0.0.0+179
AS path: I
Communities: Layer2-info: encaps:VPLS, control flags:Site-Down
Label-base: 800000, range: 8, status-vector: 0xFF
```

show route table vpn-a

```
user@host> show route table vpn-a
vpn-a.l2vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

+ = Active Route, - = Last Active, * = Both
192.168.16.1:1:1:1/96
                    *[VPN/7] 05:48:27
                       Discard
192.168.24.1:1:2:1/96
                    *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                       AS path: I
                     > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

```
192.168.24.1:1:3:1/96
                   *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table vpn-a.mdt.0

```
user@host> show route table vpn-a.mdt.0
vpn-a.mdt.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:0:10.255.14.216:232.1.1.1/144
                   *[MVPN/70] 01:23:05, metric2 1
                      Indirect
1:1:1:10.255.14.218:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.218
                      AS path: I
                    > via so-0/0/0.0, label-switched-path r0e-to-r1
1:1:2:10.255.14.217:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.217
                      AS path: I
> via so-0/0/1.0, label-switched-path r0-to-r2
```

show route table VPN-A detail

```
user@host> show route table VPN-A detail
VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
10.255.179.9/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.179.13:200
                Next hop type: Indirect
                Next-hop reference count: 5
                Source: 10.255.179.13
                Next hop type: Router, Next hop index: 732
                Next hop: 10.39.1.14 via fe-0/3/0.0, selected
                Label operation: Push 299824, Push 299824(top)
                Protocol next hop: 10.255.179.13
                Push 299824
                Indirect next hop: 8f275a0 1048574
                State: (Secondary Active Int Ext)
                Local AS: 1 Peer AS: 1
                Age: 3:41:06 Metric: 1 Metric2: 1
                Task: BGP_1.10.255.179.13+64309
                Announcement bits (2): 0-KRT 1-BGP RT Background
                AS path: I
                Communities: target:1:200 rte-type:0.0.0.0:1:0
                Import Accepted
                VPN Label: 299824 TTL Action: vrf-ttl-propagate
                Localpref: 100
                Router ID: 10.255.179.13
                Primary Routing Table bgp.l3vpn.0
```

show route table VPN-AB.inet.0

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

      10.39.1.0/30       *[OSPF/10] 00:07:24, metric 1
                          > via so-7/3/1.0
      10.39.1.4/30       *[Direct/0] 00:08:42
                          > via so-5/1/0.0
```

```
10.39.1.6/32       *[Local/0] 00:08:46
                      Local
10.255.71.16/32    *[Static/5] 00:07:24
                    > via so-2/0/0.0
10.255.71.17/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
10.255.71.15
                      AS path: I
                    > via so-2/1/0.0, Push 100020, Push 100011(top)
10.255.71.18/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
10.255.71.15
                      AS path: I
                    > via so-2/1/0.0, Push 100021, Push 100011(top)
10.255.245.245/32  *[BGP/170] 00:08:35, localpref 100
                      AS path: 2 I
                    > to 10.39.1.5 via so-5/1/0.0
10.255.245.246/32  *[OSPF/10] 00:07:24, metric 1
                    > via so-7/3/1.0
```

show route table VPN_blue.mvpn-inet6.0

```
user@host> show route table VPN_blue.mvpn-inet6.0
vpn_blue.mvpn-inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.2.202:65535:10.255.2.202/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
1:10.255.2.203:65535:10.255.2.203/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.203
                      AS path: I
                    > via so-0/1/0.0
1:10.255.2.204:65535:10.255.2.204/432
                   *[MVPN/70] 00:57:23, metric2 1
                      Indirect
5:10.255.2.202:65535:128:::192.168.90.2:128:ffff::1/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
6:10.255.2.203:65535:65000:128:::10.12.53.12:128:ffff::1/432
                   *[PIM/105] 00:02:37
                      Multicast (IPv6)
7:10.255.2.202:65535:65000:128:::192.168.90.2:128:ffff::1/432
                   *[MVPN/70] 00:02:37, metric2 1
                      Indirect
```

show route table VPN-A detail

```
user@host> show route table VPN-A detail
VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
10.255.179.9/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.179.13:200
                Next hop type: Indirect
                Next-hop reference count: 5
                Source: 10.255.179.13
                Next hop type: Router, Next hop index: 732
                Next hop: 10.39.1.14 via fe-0/3/0.0, selected
                Label operation: Push 299824, Push 299824(top)
                Protocol next hop: 10.255.179.13
                Push 299824
```

```
                    Indirect next hop: 8f275a0 1048574
                    State: (Secondary Active Int Ext)
                    Local AS: 1 Peer AS: 1
                    Age: 3:41:06 Metric: 1 Metric2: 1
                    Task: BGP_1.10.255.179.13+64309
                    Announcement bits (2): 0-KRT 1-BGP RT Background
                    AS path: I
                    Communities: target:1:200 rte-type:0.0.0.0:1:0
                    Import Accepted
                    VPN Label: 299824 TTL Action: vrf-ttl-propagate
                    Localpref: 100
                    Router ID: 10.255.179.13
                    Primary Routing Table bgp.l3vpn.0
```

show route table inetflow detail

```
user@host> show route table inetflow detail
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 2
                State: <Active Ext>
                Local AS: 65002 Peer AS: 65000
                Age: 4
                Task: BGP_65000.10.12.99.5+3792
                Announcement bits (1): 0-Flow
                AS path: 65000 I
                Communities: traffic-rate:0:0
                Validation state: Accept, Originator: 10.12.99.5
                Via: 10.12.44.0/24, Active
                Localpref: 100
                Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
        *Flow   Preference: 5
                Next-hop reference count: 2
                State: <Active>
                Local AS: 65002
                Age: 6:30
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
                AS path: I
                Communities: 1:1

user@PE1> show route table green.l2vpn.0 (VPLS Multihoming with FEC 129)
green.l2vpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.2:100:1.1.1.2/96 AD
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
1.1.1.4:100:1.1.1.4/96 AD
                   *[BGP/170] 1d 03:11:02, localpref 100, from 1.1.1.4
                      AS path: I, validation-state: unverified
                    >    via ge-1/2/1.5
1.1.1.2:100:1:0/96 MH
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
1.1.1.4:100:1:0/96 MH
                   *[BGP/170] 1d 03:11:02, localpref 100, from 1.1.1.4
                      AS path: I, validation-state: unverified
```

```
                     >    via ge-1/2/1.5
1.1.1.4:NoCtrlWord:5:100:100:1.1.1.2:1.1.1.4/176
                    *[VPLS/7] 1d 03:11:02, metric2 1
                     >    via ge-1/2/1.5
1.1.1.4:NoCtrlWord:5:100:100:1.1.1.4:1.1.1.2/176
                    *[LDP/9] 1d 03:11:02
                          Discard

user@host> show route table red extensive
red.inet.0: 364481 destinations, 714087 routes (364480 active, 48448 holddown, 1
 hidden)
22.0.0.0/32 (3 entries, 1 announced)
        State: <OnList CalcForwarding>
TSI:
KRT in-kernel 22.0.0.0/32 -> {composite(1048575)} Page 0 idx 1 Type 1 val 0x934342c

    Nexthop: Self
    AS path: [2] I
    Communities: target:2:1
Path 22.0.0.0 from 2.3.0.0 Vector len 4.  Val: 1
        @BGP    Preference: 170/-1
                Route Distinguisher: 2:1
                Next hop type: Indirect
                Address: 0x258059e4
                Next-hop reference count: 2
                Source: 2.2.0.0
                Next hop type: Router
                Next hop: 10.1.1.1 via ge-1/1/9.0, selected
                Label operation: Push 707633
                Label TTL action: prop-ttl
                Session Id: 0x17d8
                Protocol next hop: 2.2.0.0
                Push 16
                Composite next hop: 0x25805988 - INH Session ID: 0x193c
                Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                State: <Secondary Active Int Ext ProtectionPath ProtectionCand>
                Local AS:     2 Peer AS:     2
                Age: 23         Metric2: 35
                Validation State: unverified
                Task: BGP_2.2.2.0.0+34549
                AS path: I
                Communities: target:2:1
                Import Accepted
                VPN Label: 16
                Localpref: 0
                Router ID: 2.2.0.0
                Primary Routing Table bgp.l3vpn.0
                Composite next hops: 1
                        Protocol next hop: 2.2.0.0 Metric: 35
                        Push 16
                        Composite next hop: 0x25805988 - INH Session ID: 0x193c
                        Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.1.1.1 via ge-1/1/9.0
                                Session Id: 0x17d8
                        2.2.0.0/32 Originating RIB: inet.3
                          Metric: 35                      Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.1.1.1 via ge-1/1/9.0
         BGP    Preference: 170/-1
```

```
                    Route Distinguisher: 2:1
                    Next hop type: Indirect
                    Address: 0x9347028
                    Next-hop reference count: 3
                    Source: 2.3.0.0
                    Next hop type: Router, Next hop index: 702
                    Next hop: 10.1.4.2 via ge-1/0/0.0, selected
                    Label operation: Push 634278
                    Label TTL action: prop-ttl
                    Session Id: 0x17d9
                    Protocol next hop: 2.3.0.0
                    Push 16
                    Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
                    Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da
                    State: <Secondary NotBest Int Ext ProtectionPath ProtectionCand>

                    Inactive reason: Not Best in its group - IGP metric
                    Local AS:     2 Peer AS:     2
                    Age: 3:34       Metric2: 70
                    Validation State: unverified
                    Task: BGP_2.2.3.0.0+32805
                    Announcement bits (2): 0-KRT 1-BGP_RT_Background
                    AS path: I
                    Communities: target:2:1
                    Import Accepted
                    VPN Label: 16
                    Localpref: 0
                    Router ID: 2.3.0.0
                    Primary Routing Table bgp.l3vpn.0
                    Composite next hops: 1
                            Protocol next hop: 2.3.0.0 Metric: 70
                            Push 16
                            Composite next hop: 0x93463a0 1048575 INH Session ID:
0x17da
                            Indirect next hop: 0x91e8800 1048574 INH Session ID:
0x17da
                            Indirect path forwarding next hops: 1
                                    Next hop type: Router
                                    Next hop: 10.1.4.2 via ge-1/0/0.0
                                    Session Id: 0x17d9
                            2.3.0.0/32 Originating RIB: inet.3
                              Metric: 70                    Node path count: 1
                              Forwarding nexthops: 1
                                    Nexthop: 10.1.4.2 via ge-1/0/0.0
                #Multipath Preference: 255
                    Next hop type: Indirect
                    Address: 0x24afca30
                    Next-hop reference count: 1
                    Next hop type: Router
                    Next hop: 10.1.1.1 via ge-1/1/9.0, selected
                    Label operation: Push 707633
                    Label TTL action: prop-ttl
                    Session Id: 0x17d8
                    Next hop type: Router, Next hop index: 702
                    Next hop: 10.1.4.2 via ge-1/0/0.0
                    Label operation: Push 634278
                    Label TTL action: prop-ttl
                    Session Id: 0x17d9
                    Protocol next hop: 2.2.0.0
                    Push 16
                    Composite next hop: 0x25805988 - INH Session ID: 0x193c
```

```
                    Indirect next hop: 0x23eea900 - INH Session ID: 0x193c Weight 0x1

                     Protocol next hop: 2.3.0.0
                     Push 16
                     Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
                    Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da Weight
0x4000
                     State: <ForwardingOnly Int Ext>
                     Inactive reason: Forwarding use only
                     Age: 23         Metric2: 35
                     Validation State: unverified
                     Task: RT
                     AS path: I
                     Communities: target:2:1
```

show route terse

Syntax show route terse
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches) show route terse

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description Display a high-level summary of the routes in the routing table.

NOTE: For BGP routes, the **show route terse** command displays the local preference attribute and MED instead of the metric1 and metric2 values. This is mostly due to historical reasons.

To display the metric1 and metric2 value of a BGP route, use the show route extensive command.

Options **none**—Display a high-level summary of the routes in the routing table.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level view

List of Sample Output show route terse on page 648

Output Fields Table 42 on page 646 describes the output fields for the **show route terse** command. Output fields are listed in the approximate order in which they appear.

Table 42: show route terse Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number destinations	Number of destinations for which there are routes in the routing table.
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)

Table 42: show route terse Output Fields *(continued)*

Field Name	Field Description
route key	Key for the state of the route: • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route.
A	Active route. An asterisk (*) indicates this is the active route.
V	Validation status of the route: • **?**—Not evaluated. Indicates that the route was not learned through BGP. • **I**—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. • **N**—Unknown. Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **V**—Valid. Indicates that the prefix and autonomous system pair are found in the database.
Destination	Destination of the route.
P	Protocol through which the route was learned: • **A**—Aggregate • **B**—BGP • **C**—CCC • **D**—Direct • **G**—GMPLS • **I**—IS-IS • **L**—L2CKT, L2VPN, LDP, Local • **K**—Kernel • **M**—MPLS, MSDP • **O**—OSPF • **P**—PIM • **R**—RIP, RIPng • **S**—Static • **T**—Tunnel
Prf	Preference value of the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.
Metric 1	First metric value in the route. For routes learned from BGP, this is the MED metric.
Metric 2	Second metric value in the route. For routes learned from BGP, this is the IGP metric.

Table 42: show route terse Output Fields *(continued)*

Field Name	Field Description
Next hop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated.

Sample Output

show route terse

```
user@host> show route terse
inet.0: 10 destinations, 12 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 1.0.1.1/32         O  10          1             >10.0.0.2
  ?                    B 170        100                           I
  unverified                                      >10.0.0.2
* ? 1.1.1.1/32         D   0                       >lo0.2
* V 2.2.0.2/32         B 170        110                           200 I
  valid                                           >10.0.0.2
* ? 10.0.0.0/30        D   0                       >lt-1/2/0.1
  ?                    B 170        100                           I
  unverified                                      >10.0.0.2
* ? 10.0.0.1/32        L   0                        Local
* ? 10.0.0.4/30        B 170        100                           I
  unverified                                      >10.0.0.2
* ? 10.0.0.8/30        B 170        100                           I
  unverified                                      >10.0.0.2
* I 172.16.1.1/32      B 170         90                           200 I
  invalid                                         >10.0.0.2
* N 192.168.2.3/32     B 170        100                           200 I
  unknown                                         >10.0.0.2
* ? 224.0.0.5/32       O  10          1            MultiRecv
```

show validation database

Syntax

show validation database
<brief | detail>
<instance *instance-name*>
<logical-system *logical-system-name*>
<mismatch>
<origin-autonomous-system *as-number*>
<record *ip-prefix*>
<session *ip-address*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display information about the route validation database when resource public key infrastructure (RPKI) BGP route validation is configured. You can query all route validation records that match a given prefix or origin-autonomous-system. In addition, you can filter the output by a specific RPKI cache session.

Options

none—Display all route validation database entries.

brief | detail—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display information about route validation database entries for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

mismatch—(Optional) Filter the output by mismatched origin autonomous systems.

origin-autonomous-system *as-number*—(Optional) Filter the output by mismatched origin autonomous systems. The **mismatch** qualifier is useful for finding conflicting origin-autonomous-system information between RPKI caches. Mismatches might occur during cache reconfiguration.

record *ip-prefix*—(Optional) Filter the output by route validation records that match a given prefix.

session *ip-address*—(Optional) Filter the output by a specific RPKI cache session.

Required Privilege Level

view

Related Documentation

- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation database on page 650

Output Fields

Table 43 on page 650 describes the output fields for the **show validation database** command. Output fields are listed in the approximate order in which they appear.

Table 43: show validation database Output Fields

Field Name	Field Description	Level of Output
Prefix	Route validation (RV) record prefix. RV records are received from the cache server and can also be configured statically at the **[edit routing-options validation static]** hierarchy level .	All levels
Origin-AS	Legitimate originator autonomous system (AS).	All levels
Session	IP address of the RPKI cache server.	All levels
State	State of the route validation records. The state can be **valid**, **invalid** or **unknown**.	All levels
Mismatch	Conflicting origin-autonomous-system information between RPKI caches when nonstop active routing (NSR) is configured.	All levels
IPv4 records	Number of IPv4 route validation records.	All levels
IPv6 records	Number of IPv6 route validation records.	All levels

Sample Output

show validation database

```
user@host> show validation database
 RV database for instance master

  Prefix               Origin-AS Session         State   Mismatch
  1.0.1.0/24-32                1 10.0.77.1       valid
  1.0.2.0/24-32                2 10.0.77.1       valid
  1.0.3.0/24-32                3 10.0.77.1       valid
  1.0.4.0/24-32                4 10.0.77.1       valid
  1.0.5.0/24-32                5 10.0.77.1       valid
  1.0.6.0/24-32                6 10.0.77.1       valid
  1.0.7.0/24-32                7 10.0.77.1       valid
  1.0.8.0/24-32                8 10.0.77.1       valid
  72.9.224.0/19-24         26234 192.168.1.100   valid   *
  72.9.224.0/19-24          3320 192.168.1.200   invalid *
  10.0.0.0/8-32                0 internal        valid

  IPv4 records: 14
  IPv6 records: 0
```

show validation group

Syntax
show validation group
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information
Command introduced in Junos OS Release 12.2.

Description
Display information about route validation redundancy groups.

Options
none—Display information about all route validation groups.

instance *instance-name*—(Optional) Display information about route validation groups for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

Required Privilege Level
view

Related Documentation
- *Example: Configuring Origin Validation for BGP*

List of Sample Output
show validation group on page 652

Output Fields
Table 44 on page 651 describes the output fields for the **show validation group** command. Output fields are listed in the approximate order in which they appear.

Table 44: show validation group Output Fields

Field Name	Field Description
Group	Group name.
Maximum sessions	Number of concurrent sessions for each group. The default is 2. The number is configurable with the **max-sessions** statement.
Session	Resource public key infrastructure (RPKI) cache session IP address.
State	State of the connection between the routing device and the cache server. **Up** means that the connection is established. **Connect** means that the connection is not established.
Preference	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.

Sample Output

show validation group

```
user@host> show validation group
master
    Group: test, Maximum sessions: 3
      Session 10.255.255.11, State: Up, Preference: 100
      Session 10.255.255.12, State: Up, Preference: 100
    Group: test2, Maximum sessions: 2
      Session 10.255.255.13, State: Connect, Preference: 100
```

show validation replication database

Syntax

show validation replication database
<brief | detail>
<instance *instance-name*>
<logical-system *logical-system-name*>
<origin-autonomous-system *as-number*>
<record *ip-prefix*>
<session *ip-address*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display the state of the nonstop active routing (NSR) records. The output is the same as the output of the show validation database command, except for the **Mismatch** column.

Options

none—Display all route validation database entries.

brief | detail—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display information about route validation database entries for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

origin-autonomous-system *as-number*—(Optional) Filter the output by mismatched origin autonomous systems. The **mismatch** qualifier is useful for finding conflicting origin-autonomous-system information between resource public key infrastructure (RPKI) caches. Mismatches might occur during cache reconfiguration.

record *ip-prefix*—(Optional) Filter the output by route validation records that match a given prefix.

session *ip-address*—(Optional) Filter the output by a specific RPKI cache session.

Required Privilege Level

view

Related Documentation

- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation replication database on page 654

Output Fields

Table 45 on page 654 describes the output fields for the **show validation replication database** command. Output fields are listed in the approximate order in which they appear.

Table 45: show validation replication database Output Fields

Field Name	Field Description	Level of Output
Prefix	Route validation (RV) record prefix. RV records are received from the cache server and can also be configured statically at the **[edit routing-options validation static]** hierarchy level.	All levels
Origin-AS	Legitimate originator autonomous system (AS).	All levels
Session	IP address of the RPKI cache server.	All levels
State	State of the route validation records. The state can be **valid** or **invalid**.	All levels
IPv4 records	Number of IPv4 route validation records.	All levels
IPv6 records	Number of IPv6 route validation records.	All levels

Sample Output

show validation replication database

```
user@host> show validation replication database
RV database for instance master

  Prefix                Origin-AS Session         State
  1.0.1.0/24-32                 1 10.0.77.1       valid
  1.0.2.0/24-32                 2 10.0.77.1       valid
  1.0.3.0/24-32                 3 10.0.77.1       valid
  1.0.4.0/24-32                 4 10.0.77.1       valid
  1.0.5.0/24-32                 5 10.0.77.1       valid
  1.0.6.0/24-32                 6 10.0.77.1       valid
  1.0.7.0/24-32                 7 10.0.77.1       valid
  1.0.8.0/24-32                 8 10.0.77.1       valid
  72.9.224.0/19-24          26234 192.168.1.100   valid
  72.9.224.0/19-24           3320 192.168.1.200   invalid
  10.0.0.0/8-32                 0 internal        valid

  IPv4 records: 14
  IPv6 records: 0
```

show validation session

Syntax
show validation session
<brief | detail>
<*destination*>
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information
Command introduced in Junos OS Release 12.2.

Description
Display information about all sessions or a specific session with a resource public key infrastructure (RPKI) cache server.

Options
none—Display information about all sessions.

destination—(Optional) Display information about a specific session.

brief | detail—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display information about sessions for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

Required Privilege Level
view

Related Documentation
- *Example: Configuring Origin Validation for BGP*

List of Sample Output
show validation session brief on page 657
show validation session detail on page 657

Output Fields
Table 46 on page 655 describes the output fields for the **show validation session** command. Output fields are listed in the approximate order in which they appear.

Table 46: show validation session Output Fields

Field Name	Field Description	Level of Output
Session	IP address of the RPKI cache server.	All levels
State	State of the connection between the routing device and the cache server. **Up** means that the connection is established. **Connect** means that the connection is not established.	All levels
Flaps	Number of attempts to establish a session.	None and **brief**

Table 46: show validation session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Uptime	Length of time that the session has remained establshed.	None and **brief**
#IPv4/IPv6 records	Number of IPv4 and IPv6 route validation records.	None and **brief**
Session index	Every session has an index number.	**detail**
Group	Name of the group to which the session belongs	**detail**
Preference	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.	**detail**
Port	TCP port number for the outgoing connection with the cache server. The well-known RPKI port is TCP port 2222. For a given deployment, an RPKI cache server might listen on some other TCP port number. If so, you can configure the alternative port number with the **port** statement.	**detail**
Refresh time	Liveliness check interval for an RPKI cache server. Every**refresh-time** (seconds), a serial query protocol data unit (PDU) with the last known serial number is transmitted. The **hold-time** must be at least 2 x the **refresh-time**.	**detail**
Hold time	Length of time in seconds that the session between the routing device and the cache server is considered operational without any activity. After the hold time expires, the session is dropped. Reception of any PDU from the cache server resets the hold timer. The **hold-time** is 600 seconds, by default, and must be least 2 x the **refresh-time**. If the hold time expires, the session is considered to be down. This, in turn, triggers a session restart event. During a session restart, the routing device attempts to start a session with the cache server that has the numerically highest **preference**.	**detail**
Record Life time	Amount of time that route validation (RV) records learned from a cache server are valid. RV records expire if the session to the cache server goes down and remains down for the **record-lifetime** (seconds).	**detail**
Serial (Full Update)	Number of full serial updates.	**detail**
Serial (Incremental Update)	Number of incremental serial updates.	**detail**
Session flaps	Number of attempts to establish a session.	**detail**

Table 46: show validation session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Session uptime	Length of time that the session has remained establshed.	**detail**
Last PDU received	Time when the most recent PDU was received.	**detail**
IPv4 prefix count	Number of IPv4 sessions.	**detail**
IPv6 prefix count	Number of IPv6 sessions.	**detail**

Sample Output

show validation session brief

```
user@host> show validation session brief
 Session                                State    Flaps     Uptime #IPv4/IPv6
records
  1.3.0.2                               up           2   00:01:37 13/0
  10.255.255.11                         up           3   00:00:01 1/0
  10.255.255.12                         connect      2            64/68
```

show validation session detail

```
user@host> show validation session detail
Session 10.0.77.1, State: up
    Group: test, Preference: 100
    Local IPv4 address: 10.0.77.2, Port: 2222
    Refresh time: 300s
    Session flaps: 14, Last Session flap: 5h13m18s ago
    Hold time: 900s
    Record Life time: 3600s
    Serial (Full Update): 0
    Serial (Incremental Update): 0
      Session flaps 2
      Session uptime: 00:48:35
      Last PDU received: 00:03:35
      IPv4 prefix count: 71234
      IPv6 prefix count: 345
```

show validation statistics

Syntax show validation statistics
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information Command introduced in Junos OS Release 12.2.

Description Display route validation statistics.

Options **none**—Display statistics for all routing instances.

instance ***instance-name***—(Optional) Display information for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system ***logical-system-name***—(Optional) Perform this operation on a particular logical system.

Required Privilege Level view

Related Documentation
- *Example: Configuring Origin Validation for BGP*

List of Sample Output show validation statistics on page 659

Output Fields Table 47 on page 658 describes the output fields for the **show validation statistics** command. Output fields are listed in the approximate order in which they appear.

Table 47: show validation statistics Output Fields

Field Name	Field Description
Total RV records	Group name.
Total Replication RV records	Number of concurrent sessions for each group. The default is 2. The number is configurable with the **max-sessions** statement.
Prefix entries	Resource public key infrastructure (RPKI) cache session IP address.
Origin-AS entries	State of the connection between the routing device and the cache server. **Up** means that the connection is up. **Connect** means that the connection is not up.
Memory utilization	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.

Table 47: show validation statistics Output Fields *(continued)*

Field Name	Field Description
Policy origin-validation requests	Number of queries for validation state of a given instance and prefix.
Valid	Number of valid prefixes reported by the validation query.
Invalid	Number of invalid prefixes reported by the validation query.
Unknown	Number of unknown prefixes reported by the validation query. This means that the prefix is not found in the database.
BGP import policy reevaluation notifications	A change, addition, or deletion of a route validation record triggers a BGP import reevaluation for all exact matching and more specific prefixes.
inet.0	Number of IPv4 route validation records that have been added, deleted, or changed.
inet6.0	Number of IPv6 route validation records that have been added, deleted, or changed.

Sample Output

show validation statistics

```
user@host> show validation statistics
Total RV records:        453455
  Total Replication RV records: 453455
    Prefix entries:         35432
    Origin-AS entries:     124400
  Memory utilization: 16.31MB
  Policy origin-validation requests: 234995
    valid:                  23445
    invalid:                14666
    unknown:                34567
  BGP import policy reevaluation notifications: 460268
    inet.0:                435345
    inet6.0:                 3454
```

test policy

Syntax test policy *policy-name prefix*

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description Test a policy configuration to determine which prefixes match routes in the routing table.

NOTE: If you are using the **test policy** command on a logical system, you must first set the CLI to the logical system context. For example, if you want to test a routing policy that is configured on logical system R2, first run the **set cli logical-system R2** command.

Options ***policy-name***—Name of a policy.

prefix—Destination prefix to match.

Additional Information All prefixes in the default unicast routing table (inet.0) that match prefixes that are the same as or longer than the specific prefix are processed by the **from** clause in the specified policy. All prefixes accepted by the policy are displayed. The **test policy** command evaluates a policy differently from the BGP import process. When testing a policy that contains an **interface** match condition in the **from** clause, the **test policy** command uses the match condition. In contrast, BGP does not use the **interface** match condition when evaluating the policy against routes learned from internal BGP (IBGP) or external BGP (EGBP) multihop peers.

Required Privilege Level view

Related Documentation
- Understanding Routing Policy Tests on page 79
- Example: Testing a Routing Policy with Complex Regular Expressions on page 367

List of Sample Output test policy on page 660

Output Fields For information about output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

test policy

```
user@host> test policy test-statics 3.0.0.1/8
inet.0: 44 destinations, 44 routes (44 active, 0 holddown, 0 hidden)
Prefixes passing policy:
```

```
3.0.0.0/8          *[BGP/170] 16:22:46, localpref 100, from 10.255.255.41
                   AS Path: 50888 I
                    > to 10.11.4.32 via en0.2, label-switched-path l2
3.3.3.1/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                    > to 10.0.4.7 via fxp0.0
3.3.3.2/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                    > to 10.0.4.7 via fxp0.0
3.3.3.3/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                    > to 10.0.4.7 via fxp0.0
3.3.3.4/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                    > to 10.0.4.7 via fxp0.0
Policy test-statics: 5 prefixes accepted, 0 prefixes rejected
```

PART 4

Index

- Index on page 665

Index

Symbols

!
  in policy expressions
    logical operator....20
#, comments in configuration statements....xviii
&&, logical operator....19
( ), in syntax descriptions....xviii
< >, in syntax descriptions....xviii
[ ], in configuration statements....xviii
{ }, in configuration statements....xviii
| (pipe), in syntax descriptions....xviii
|| (pipes), logical operator....20

A

accept
  policy, routing
    control action....121
actions
  policy, routing
    characteristics, manipulating....121
    flow control....119, 121
    tracing....119
  route list match types....116
  routing policy, summary of....118
  tracing....129
add-path statement
  BGP
    usage guidelines....301
address-family statement....445
address-mask (route filter match type)....29
  evaluation in a route filter....34
  example....40, 41, 42
administrative distance
  BGP *See* preference statement
advertise-external statement
  usage guidelines....204
advertise-inactive statement
  usage guidelines....212
advertise-peer-as statement
  usage guidelines....197
advertisements, displaying 508
aigp statement
  BGP....446
apply-path statement....447
  usage guidelines....46
area (routing policy match condition)....110
AS path
  prepending....292
AS paths
  matching regular expressions, displaying....515
as-path (routing policy match condition)....110
as-path statement....448
  policy, routing
    usage guidelines....57
as-path-group statement....449
  usage guidelines....57
as-path-prepend (routing policy action)....63, 121
ASN
  BGP community routes, displaying....522
ASs
  paths
    modifying with routing policy....63, 121
  regular expressions *See* policy, routing, AS path regular expressions
autonomous system number *See* ASN

B

best routes, displaying....517
BGP
  advertise-peer-as....197
  advertising multiple paths to a destination....301
  applying routing policies....133
  best external route
    advertising....204
  communities
    names....68
    policy, routing....67, 451
  community ASN, displaying routes....522
  community name, displaying routes....524
  community remove....358
  community-count....351
  damping parameters....82, 455
    displaying....496
  damping routes, displaying....526
  dynamic routing policies
    applying....100
  extended communities....72
  import policy
    family qualifier for....257

MED....245
policy, routing....458, 467
applying....130
preferences....223
proxy route target filtering....486
route validation
information,
displaying....649, 651, 653, 655, 658
routing tables
nonactive routes....212
BGP (Border Gateway Protocol)
policy to make routes less preferable....292
route-flap damping....81
BGP confederations
route-flap damping....81
braces, in configuration statements....xviii
brackets
angle, in syntax descriptions....xviii
square, in configuration statements....xviii
BSR
policy, import....473

C

ccc statement....450
class (routing policy action)....122
color
policy, routing
action....122
match condition....110
comments, in configuration statements....xviii
communities
extend range of BGP communities....72
names....68
policy, routing....67, 451
action....122
match condition....110
community ASN, displaying routes....522
community name, displaying routes....524
community remove....358
community statement....451
policy, routing
usage guidelines....68
community-count match condition....351
condition statement....454
conditions
routing policy....494
conventions
text and syntax....xvii
curly braces, in configuration statements....xviii
customer support....xix
contacting JTAC....xix

D

damping
policy, routing, action....122
damping parameters, BGP
displaying....496
damping routes, BGP
displaying....526
damping statement....455
BGP
usage guidelines....85
policy, routing
usage guidelines....83
usage guidelines....375, 384
default route
conditionalizing....142
destination class usage....123, 124
destination-class (routing policy action)....123, 126
discard interface....103
described....103
documentation
comments on....xix
dsc interface....103
described....103
See also discard interface
DVMRP
policy, routing....460, 468
applying....130
dynamic database
active nonstop routing....100
routing policies....97
dynamic routing policies
active nonstop routing....100
BGP....100
configuring....98
dynamic-db statement....456
overview....421, 435
dynamic-db statement....456
usage guidelines....98

E

EBGP (external BGP)
route-flap damping....81
exact (route filter match type)....29
exact route list match type....48, 116
export route information, displaying....549

export statement....460
BGP....458
DVMRP....460
forwarding table....466
IS-IS....459
MSDP....461
OSPF....462
PIM....463
policy, routing
usage guidelines....23
RIP....464
RIPng....465

F
fault tolerance
advertising multiple paths to a destination....301
filtering received labels....469
firewall filters
comparison with routing policies....8, 11
flow, packets....4
purpose....8
flap damping....81
parameters....81
flooding
IS-IS and OSPF....16
flow control actions....119, 121
flow control, actions in routing policies....118
font conventions....xvii
forwarding classes
policy to group source and destination prefixes....395
forwarding table
policy, routing....466
route entries, displaying....570
from statement....480
policy, routing
usage guidelines....107

G
generate statement
usage guidelines....142

H
hidden routes, displaying....584

I
if-route-exists statement....454, 457
import routing policies
applying....19
import statement
BGP....467
bootstrap....473
DVMRP....468
LDP....469
MSDP....470
OSPF....471
PIM....472
RIP....474
RIPng....475
route resolution....476
inet statement....476
inet-vpn statement
usage guidelines....257
inet6-vpn statement
usage guidelines....257
install-nexthop lsp (routing policy action)....123
instance (routing policy match condition)....111
interface (routing policy match condition)....111
invert-match statement
usage guidelines....76
IS-IS
policy, routing....459
applying....130

J
joins, PIM
rejecting....39

L
label filtering....469
LDP
policy filters....469
policy, routing
applying....130
received label filtering....469
level (routing policy match condition)....111
load balancing
advertising multiple paths to a destination....301
local-preference
policy, routing
action....123
match condition....111
longer (route filter match type)....29
longer route list match type....48, 116

M

manuals
  comments on....xix
match conditions
  policy, routing....105, 107
  routing policy, summary of....108
match types....116
MBGP MVPNs....384
MED *See* BGP
metric
  policy, routing
    action....124
    match condition....112
metric statement
  BGP
    usage guidelines....245
MPLS
  policy, routing
    applying....130
MSDP
  policy, routing....461, 470
multicast-scoping
  policy, routing
    match condition....112

N

neighbor (routing policy match condition)....112
next hops
  routes sent to, displaying....599
next policy (routing policy control action)....121
next term (routing policy control action)....121
next-hop
  policy, routing
    action....125
    match condition....112
nlri-route-type statement
  usage guidelines....384
no-advertise-peer-as statement
  usage guidelines....197

O

origin
  policy, routing
    action....125
    match condition....113
orlonger (route filter match type)....30
orlonger route list match type....48, 116
OSPF
  default route policy....142
  policy, routing....462, 471
    applying....130
    route install priority....240

P

parentheses, in syntax descriptions....xviii
path-count statement
  BGP
    usage guidelines....301
peer-unit statement....477
PIM
  multicast traffic joins, rejecting....39
  policy, routing....472
    applying....130
policy (routing policy match condition)....113
policy chain....229, 283
policy filters, LDP....469
policy framework
  comparison of policies....8
  firewall filters....3
  overview....3
  policy, routing....3
policy, import
  BSR....473
policy, routing
  actions....119, 123, 124, 129
  AS path regular
    expressions....57, 63, 447, 448, 449
  BGP....458, 467
  BGP damping parameters....455
  chains
    evaluation....49
  communities....67, 451
  comparison with firewall filters....8
  configuring....80
  default policies and actions....15
  DVMRP....460, 468
  flow, routing information....4
  forwarding table....466
  framework....12
  import policies....23
  IS-IS....459
  match conditions....105, 107, 115
  MSDP....461, 470
  multiple policies
    evaluation....49
  OSPF....462, 471

overview....12
PIM....472
policy chain....229, 283
policy expressions....19, 23
preferences, modifying....126
prefix list....45, 261, 484
prefix list filter....47, 485
purpose....8
rejecting PIM multicast traffic joins....39
RIP....464, 474
RIPng....465, 475
route filters....25, 39
route target prefix list....486
subroutine....149, 273, 327, 342
subroutines....51, 54
uses for....4
policy-options statement....478
policy-statement statement....480
from statement....107
then statement....119
to statement....107
preference statement
BGP
usage guidelines....223
preferences
modifying
with routing policies....126
policy, routing
action....126
match condition....113
prefix list....45, 261, 484
prefix list filter....485
prefix-length-range (route filter match type)....30
prefix-length-range match type....116
prefix-list (routing policy match condition)....113
prefix-list statement....484
usage guidelines....46
prefix-list-filter statement....485
prefix-policy statement
BGP
usage guidelines....301
propagation, suppressing....81
protocols
match condition
policy, routing....113
routing
applying policies....23

R

receive statement
BGP
usage guidelines....301
received label filtering....469
regular expressions
AS paths, displaying matching routes....515
reject
policy, routing
control action....121
resource public key infrastructure *See* RPKI
rib (routing policy match condition)....114
RIP
policy, routing....464, 474
applying....130
RIPng
policy, routing....465, 475
route
generate statement
usage guidelines....142
route advertisements, displaying....508
route filters....25
route injection....240
route list match types....116
route manipulation actions, routing policies....118
route redistribution....240
route target prefix list....486
route, displaying
next-hop....599
route-filter (routing policy match condition)....114
route-filter statement
usage guidelines....257
route-flap damping....81
parameters....81
routes, displaying
active....498
active path....503
advertising protocol....508
all....513
AS paths
regular expressions, matching....515
best....517
brief information....520
community ASN....522
community name....524
damping, BGP....526
detailed information....531
extensive information....552
flow validation....568

hidden....584
in a specific routing table....633
in the forwarding table....570
inactive path....587
inactive prefix....590
instances....592
learned from a specific protocol....613
matching the specified address....547
not associated with a community....605
policy-based route export....549
received through a neighbor....625
sent to a specific interface....608
terse information....646
routing policies
configuration tasks....240, 292, 375, 395
displaying....492
dynamic
configuring....98
dynamic database....97
forwarding class with source and destination....395
grouping source and destination prefixes....395
making BGP routes less preferable....292
OSPF import policy....240
prepending AS paths....292
reducing update messages with flap damping....81
route redistribution....240
route-flap damping....81
testing the configuration for....660
routing policy *See* policy, routing
applying to BGP....133
testing....367
routing solutions
making BGP routes less preferable....292
reducing update messages with flap damping....81
routing tables
nonactive routes, exchanging with BGP....212
RPKI
information, displaying....649, 651, 653, 655, 658
rtf-prefix-list statement....486

S

send statement
BGP
usage guidelines....301
show policy command....492
show policy conditions command....494
show policy damping command....87, 496
usage guidelines....83
show route active-path command....503
show route advertising-protocol command....508
show route all command....513
show route aspath-regex command....515
show route best command....517
show route brief command....520
show route command....498
show route community command....522
show route community-name command....524
show route damping command....526
show route detail command....531
usage guidelines....83
show route exact command....547
show route export command....549
show route extensive command....552
show route flow validation command....568
show route forwarding-table command....570
show route hidden command....584
show route inactive-path command....587
show route inactive-prefix command....590
show route instance command....592
show route next-hop command....599
show route no-community command....605
show route output command....608
show route protocol command....613
show route receive-protocol command....625
usage guidelines....119
show route table command....633
show route terse command....646
show validation database command....649
show validation group command....651
show validation replication database command....653
show validation session command....655
show validation statistics command....658
source class usage....126
source-address-filter (routing policy match condition)....114
source-class (routing policy action)....126
standby statement....487
static route
configuring a default....142
subroutine in a routing policy....149, 273, 327, 342
subroutines....51, 54
support, technical *See* technical support
syntax conventions....xvii

T

table statement....454, 488
tag
    policy, routing
        action....127
technical support
    contacting JTAC....xix
test policy command....367, 660
testing a routing policy....367
then statement....480
    policy, routing
        usage guidelines....119
through (route filter match type)....30
through route list match type....116
to statement....480
    usage guidelines....107
trace (policy tracing action)....119, 129
tracing actions....119, 129

U

upto (route filter match type)....30
upto route list match type....116

V

verification
    IS-IS policy....139
    network interfaces....321
    OSPF policy....147