Juniper Networks | Engineering Simplicity

Junos® OS

RIP User Guide

Published
2025-12-15

JUNOS

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

Junos® OS RIP User Guide

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About This Guide

Use this guide to configure, monitor, and troubleshoot the RIP routing protocol on your Juniper Network devices.

CHAPTER 1

Overview

IN THIS CHAPTER

RIP and RIPng Overview

IN THIS SECTION

RIP Overview

IN THIS SECTION

RIP is an interior gateway protocol (IGP) that uses a distance-vector algorithm to determine the best route to a destination, using the hop count as the metric.

In a RIP network, each router's forwarding table is distributed among the nodes through the flooding of routing table information. Because topology changes are flooded throughout the network, every node maintains the same list of destinations. Packets are then routed to these destinations based on path-cost calculations done at each node in the network.

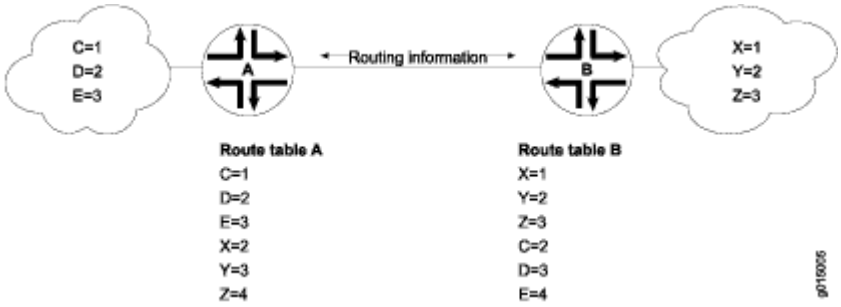
NOTE: In general, the term *RIP* refers to RIP version 1 and RIP version 2.

This topic contains the following sections:

Distance-Vector Routing Protocols

Distance-vector routing protocols transmit routing information that includes a distance vector, typically expressed as the number of hops to the destination. This information is flooded out all protocol-enabled interfaces at regular intervals (every 30 seconds in the case of RIP) to create a network map that is stored in each node's local topology database. Figure 1 on page 3 shows how distance-vector routing works.

Figure 1: Distance-Vector Protocol



In Figure 1 on page 3, Routers A and B have RIP enabled on adjacent interfaces. Router A has known RIP neighbors Routers C, D, and E, which are 1, 2, and 3 hops away, respectively. Router B has known RIP neighbors Routers X, Y, and Z, which are 1, 2, and 3 hops away, respectively. Every 30 seconds, each router floods its entire routing table information out all RIP-enabled interfaces. In this case, flooding exchanges routing table information across the RIP link.

When Router A receives routing information from Router B, it adds 1 to the hop count to determine the new hop count. For example, Router X has a hop count of 1, but when Router A imports the route to X, the new hop count is 2. The imported route also includes information about where the route was learned, so that the original route is imported as a route to Router X through Router B with a hop count of 2.

When multiple routes to the same host are received, RIP uses the distance-vector algorithm to determine which path to import into the forwarding table. The route with the smallest hop count is imported. If there are multiple routes with the same hop count, all are imported into the forwarding table, and traffic is sent along the paths in round-robin fashion.

RIP Protocol Overview

The RIP IGP uses the Bellman-Ford, or *distance-vector*, algorithm to determine the best route to a destination. RIP uses the hop count as the metric. RIP enables hosts and routers to exchange information for computing routes through an IP-based network. RIP is intended to be used as an IGP in reasonably homogeneous networks of moderate size.

The Junos® operating system (Junos OS) supports RIP versions 1 and 2.

NOTE: RIP is not supported for multipoint interfaces.

RIP version 1 packets contain the minimal information necessary to route packets through a network. However, this version of RIP does not support authentication or subnetting.

RIP uses User Datagram Protocol (UDP) port 520.

RIP has the following architectural limitations:

- The longest network path cannot exceed 15 hops (assuming that each network, or hop, has a cost of 1).
- RIP depends on counting to infinity to resolve certain unusual situations—When the network consists of several hundred routers, and when a routing loop has formed, the amount of time and network bandwidth required to resolve a next hop might be great.
- RIP uses only a fixed metric to select a route. Other IGPs use additional parameters, such as measured delay, reliability, and load.

RIP Packets

RIP packets contain the following fields:

- Command—Indicates whether the packet is a request or response message. Request messages seek information for the router's routing table. Response messages are sent periodically and also when a request message is received. Periodic response messages are called *update messages*. Update messages contain the command and version fields and 25 destinations (by default), each of which includes the destination IP address and the metric to reach that destination.

  NOTE: Beginning with Junos OS Release 11.1, three additional command field types are available to support RIP demand circuits. When you configure an interface for RIP demand circuits, the command field indicates whether the packet is an update request, update response, or update acknowledge message. Neighbor interfaces send updates on demand, not periodically. These command field types are only valid on interfaces configured for RIP demand circuits. For more detailed information, see "RIP Demand Circuits Overview" on page 54.

- Version number—Version of RIP that the originating router is running.
- Address family identifier—Address family used by the originating router. The family is always IP.
- Address—IP address included in the packet.

- Metric—Value of the metric advertised for the address.
- Mask—Mask associated with the IP address (RIP version 2 only).
- Next hop—IP address of the next-hop router (RIP version 2 only).

Routing information is exchanged in a RIP network by RIP request and RIP response packets. A router that has just booted can broadcast a RIP request on all RIP-enabled interfaces. Any routers running RIP on those links receive the request and respond by sending a RIP response packet immediately to the router. The response packet contains the routing table information required to build the local copy of the network topology map.

In the absence of RIP request packets, all RIP routers broadcast a RIP response packet every 30 seconds on all RIP-enabled interfaces. The RIP broadcast is the primary way in which topology information is flooded throughout the network.

Once a router learns about a particular destination through RIP, it starts a timer. Every time it receives a new response packet with information about the destination, the router resets the timer to zero. However, if the router receives no updates about a particular destination for 180 seconds, it removes the destination from its RIP routing table.

In addition to the regular transmission of RIP packets every 30 seconds, if a router detects a new neighbor or detects that an interface is unavailable, it generates a triggered update. The new routing information is immediately broadcast out all RIP-enabled interfaces, and the change is reflected in all subsequent RIP response packets.

Maximizing Hop Count

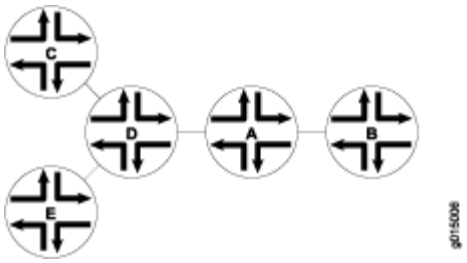
The successful routing of traffic across a RIP network requires that every node in the network maintain the same view of the topology. Topology information is broadcast between RIP neighbors every 30 seconds. If Router A is many hops away from a new host, Router B, the route to B might take significant time to propagate through the network and be imported into Router A's routing table. If the two routers are 5 hops away from each other, Router A cannot import the route to Router B until 2.5 minutes after Router B is online (30 seconds per hop). For large numbers of hops, the delay becomes prohibitive. To help prevent this delay from growing arbitrarily large, RIP enforces a maximum hop count of 15 hops. Any prefix that is more than 15 hops away is treated as unreachable and assigned a hop count equal to infinity. This maximum hop count is called the *network diameter*.

Split Horizon and Poison Reverse Efficiency Techniques

Because RIP functions by periodically flooding the entire routing table out to the network, it generates a lot of traffic. The split horizon and poison reverse techniques can help reduce the amount of network traffic originated by RIP hosts and make the transmission of routing information more efficient.

If a router receives a set of route advertisements on a particular interface, RIP determines that those advertisements do not need to be retransmitted out the same interface. This technique, known as *split horizon*, helps limit the amount of RIP routing traffic by eliminating information that other neighbors on that interface have already learned. Figure 2 on page 6 shows an example of the split horizon technique.

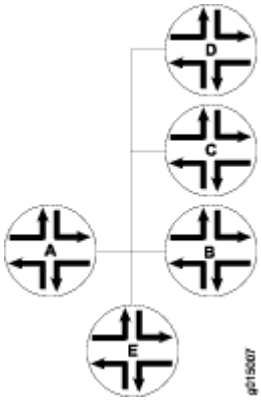
Figure 2: Split Horizon Example



In Figure 2 on page 6, Router A advertises routes to Routers C, D, and E to Router B. In this example, Router A can reach Router C in 2 hops. When Router A advertises the route to Router B, Router B imports it as a route to Router C through Router A in 3 hops. If Router B then readvertised this route to Router A, Router A would import it as a route to Router C through Router B in 4 hops. However, the advertisement from Router B to Router A is unnecessary, because Router A can already reach the route in 2 hops. The split horizon technique helps reduce extra traffic by eliminating this type of route advertisement.

Similarly, the poison reverse technique helps to optimize the transmission of routing information and improve the time to reach network convergence. If Router A learns about unreachable routes through one of its interfaces, it advertises those routes as unreachable (hop count of 16) out the same interface. Figure 3 on page 6 shows an example of the poison reverse technique.

Figure 3: Poison Reverse Example

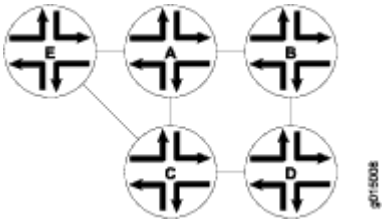


In Figure 3 on page 6, Router A learns through one of its interfaces that routes to Routers C, D, and E are unreachable. Router A readvertises those routes out the same interface as unreachable. The advertisement informs Router B that Routers C, D, and E are definitely not reachable through Router A.

Limitations of Unidirectional Connectivity

Because RIP processes routing information based solely on the receipt of routing table updates, it cannot ensure bidirectional connectivity. As Figure 4 on page 7 shows, RIP networks are limited by their unidirectional connectivity.

Figure 4: Limitations of Unidirectional Connectivity



In Figure 4 on page 7, Routers A and D flood their routing table information to Router B. Because the path to Router E has the fewest hops when routed through Router A, that route is imported into Router B's forwarding table. However, suppose that Router A can transmit traffic but is not receiving traffic from Router B because of an unavailable link or invalid routing policy. If the only route to Router E is through Router A, any traffic destined for Router A is lost, because bidirectional connectivity was never established.

OSPF establishes bidirectional connectivity with a three-way handshake.

SEE ALSO

RIP Configuration Overview | 12

Example: Configuring RIP

RIPng Overview

IN THIS SECTION

The Routing Information Protocol next generation (RIPng) is an interior gateway protocol (IGP) that uses a distance-vector algorithm to determine the best route to a destination, using hop count as the metric. RIPng exchanges routing information used to compute routes and is intended for IP version 6 (IPv6)-based networks. RIPng is disabled by default.

On devices in secure context, IPv6 is disabled. You must enable IPv6 to use RIPng. For instructions, see the *Junos OS Interfaces Configuration Guide for Security Devices*.

This topic contains the following sections:

RIPng Protocol Overview

The RIPng IGP uses the Bellman-Ford distance-vector algorithm to determine the best route to a destination, using hop count as the metric. RIPng allows hosts and routers to exchange information for computing routes through an IP-based network. RIPng is intended to act as an IGP for moderately-sized autonomous systems.

RIPng is a distinct routing protocol from RIPv2. The Junos OS implementation of RIPng is similar to RIPv2, but has the following differences:

- RIPng does not need to implement authentication on packets.
- Junos OS does not support multiple instances of RIPng.
- Junos OS does not support RIPng routing table groups.

RIPng is a UDP-based protocol and uses UDP port 521.

RIPng has the following architectural limitations:

- The longest network path cannot exceed 15 hops (assuming that each network, or hop, has a cost of 1).

- RIPng is prone to routing loops when the routing tables are reconstructed. Especially when RIPng is implemented in large networks that consist of several hundred routers, RIPng might take an extremely long time to resolve routing loops.
- RIPng uses only a fixed metric to select a route. Other IGPs use additional parameters, such as measured delay, reliability, and load.

RIPng Standards

RIPng is defined in the following documents:

- RFC 2080, *RIPng for IPv6*
- RFC 2081, *RIPng Protocol Applicability Statement*

To access Internet Requests for Comments (RFCs) and drafts, see the Internet Engineering Task Force (IETF) website.

RIPng Packets

A RIPng packet header contains the following fields:

- Command—Indicates whether the packet is a request or response message. Request messages seek information for the router's routing table. Response messages are sent periodically or when a request message is received. Periodic response messages are called update messages. Update messages contain the command and version fields and a set of destinations and metrics.
- Version number—Specifies the version of RIPng that the originating router is running. This is currently set to Version 1.

The rest of the RIPng packet contains a list of routing table entries consisting of the following fields:

- Destination prefix—128-bit IPv6 address prefix for the destination.
- Prefix length—Number of significant bits in the prefix.
- Metric—Value of the metric advertised for the address.
- Route tag—A route attribute that must be advertised and redistributed with the route. Primarily, the route tag distinguishes external RIPng routes from internal RIPng routes when routes must be redistributed across an exterior gateway protocol (EGP).

SEE ALSO

Example: Configuring a Basic RIPng Network | 134

Supported RIP and RIPng Standards

Junos OS substantially supports the following RFCs, which define standards for RIP (for IP version 4 [IPv4]) and RIP next generation (RIPng, for IP version 6 [IPv6]).

Junos OS supports authentication for all RIP protocol exchanges (MD5 or simple authentication).

- RFC 1058, *Routing Information Protocol*
- RFC 2080, *RIPng for IPv6*
- RFC 2082, *RIP-2 MD5 Authentication*

  Multiple keys using distinct key IDs are not supported.
- RFC 2453, *RIP Version 2*

The following RFC does not define a standard, but provides information about RIPng. The IETF classifies it as “Informational.”

- RFC 2081, *RIPng Protocol Applicability Statement*

SEE ALSO

Supported IPv4, TCP, and UDP Standards

Supported IPv6 Standards

Accessing Standards Documents on the Internet

2

CHAPTER

Configuring RIP

IN THIS CHAPTER

Basic RIP Configuration

IN THIS SECTION

RIP Configuration Overview

To achieve basic connectivity between all RIP hosts in a RIP network, you enable RIP on every interface that is expected to transmit and receive RIP traffic, as described in the steps that follow.

To configure a RIP network:

1. Configure network interfaces. See the *Junos OS Interfaces Configuration Guide for Security Devices*.
2. Define RIP groups, which are logical groupings of interfaces, and add interfaces to the groups. Then, configure a routing policy to export directly connected routes and routes learned through RIP routing exchanges. See "Example: Configuring a Basic RIP Network" on page 13.
3. (Optional) Configure metrics to control traffic through the RIP network. See "Example: Controlling Traffic in a RIP Network with an Incoming Metric" on page 85 and "Example: Controlling Traffic in a RIP Network with an Outgoing Metric" on page 87.
4. (Optional) Configure authentication to ensure that only trusted routers participate in the autonomous system's routing. See "Enabling Authentication with Plain-Text Passwords" on page 24 and Enabling Authentication with MD5 Authentication (CLI Procedure).

SEE ALSO

RIP Overview | 2

Verifying a RIP Configuration | 216

Understanding Basic RIP Routing

RIP is an interior gateway protocol (IGP) that routes packets within a single autonomous system (AS). By default, RIP does not advertise the subnets that are directly connected through the device's interfaces. For traffic to pass through a RIP network, you must create a routing policy to export these routes. Advertising only the direct routes propagates the routes to the immediately adjacent RIP-enabled router only. To propagate all routes through the entire RIP network, you must configure the routing policy to export the routes learned through RIP.

SEE ALSO

RIP Overview | 2

Example: Configuring a Basic RIP Network

IN THIS SECTION

This example shows how to configure a basic RIP network.

Requirements

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

Overview

IN THIS SECTION

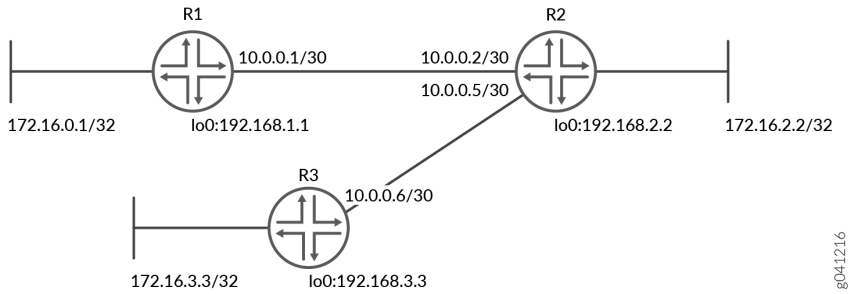
In this example, you configure a basic RIP network, create a RIP group called **rip-group**, and add the directly connected interfaces to the RIP group. Then you configure a routing policy to advertise direct routes using policy statement **advertise-routes-through-rip**.

By default, Junos OS does not advertise RIP routes, not even routes that are learned through RIP. To advertise RIP routes, you must configure and apply an export routing policy that advertises RIP-learned and direct routes.

In Junos OS, you do not need to configure the RIP version. RIP version 2 is used by default.

To use RIP on the device, you must configure RIP on all of the RIP interfaces within the network. Figure 5 on page 14 shows the topology used in this example.

Figure 5: Sample RIP Network Topology



"CLI Quick Configuration" on page 15 shows the configuration for all of the devices in Figure 5 on page 14. The section "No Link Title" on page 16 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
```

```
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 a basic RIP network:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
   user@R1# set lo0 unit 1 family inet address 172.16.0.1/32
   user@R1# set lo0 unit 1 family inet address 192.168.1.1/32
   ```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

   ```
   [edit protocols rip group rip-group]
   user@R1# set neighbor fe-1/2/0.1
   ```

3. Create the routing policy to advertise both direct and RIP-learned routes.

   ```
   [edit policy-options policy-statement advertise-routes-through-rip term 1]
   user@R1# set from protocol direct
   user@R1# set from protocol rip
   user@R1# set then accept
   ```

4. Apply the routing policy.

In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.0.1/32;
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
rip {
    group rip-group {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
```

```
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the Routing Table

Purpose

Verify that the routing table is populated with the expected routes..

Action

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

```
user@R1> show route protocol rip
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[RIP/100] 00:59:15, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
```

```
172.16.2.2/32      *[RIP/100] 02:52:48, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
172.16.3.3/32      *[RIP/100] 00:45:05, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.2.2/32     *[RIP/100] 02:52:48, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.3.3/32     *[RIP/100] 00:45:05, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
224.0.0.9/32       *[RIP/100] 00:45:09, metric 1
                          MultiRecv
```

Meaning

The output shows that the routes have been learned from Device R2 and Device R3.

If you were to delete the **from protocol rip** condition in the routing policy on Device R2, the remote routes from Device R3 would not be learned on Device R1.

Looking at the Routes That Device R1 Is Advertising to Device R2

Purpose

Verify that Device R1 is sending the expected routes.

Action

From operational mode, enter the `show route advertising-protocol rip` command.

```
user@R1> show route advertising-protocol rip 10.0.0.1
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.1/32      *[Direct/0] 05:18:26
                    >    via lo0.1
192.168.1.1/32     *[Direct/0] 05:18:25
                    >    via lo0.1
```

Meaning

Device R1 is sending routes to its directly connected networks.

Looking at the Routes That Device R1 Is Receiving from Device R2

Purpose

Verify that Device R1 is receiving the expected routes.

Action

From operational mode, enter the `show route receive-protocol rip` command.

```
user@R1> show route receive-protocol rip 10.0.0.2
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[RIP/100] 02:31:22, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
172.16.2.2/32      *[RIP/100] 04:24:55, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
172.16.3.3/32      *[RIP/100] 02:17:12, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.2.2/32     *[RIP/100] 04:24:55, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.3.3/32     *[RIP/100] 02:17:12, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
```

Meaning

Device R1 is receiving from Device R2 all of Device R2's directly connected networks. Device R1 is also receiving from Device R2 all of Device R3's directly connected networks, which Device R2 learned from Device R3 through RIP.

Verifying the RIP-Enabled Interfaces

Purpose

Verify that all RIP-enabled Interfaces are available and active.

Action

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

```
user@R1> show rip neighbor
                 Local  Source          Destination     Send   Receive   In
Neighbor         State  Address         Address         Mode   Mode     Met
--------         -----  -------         -----------     ----   -------  ---
fe-1/2/0.1          Up 10.0.0.1        224.0.0.9       mcast  both     1
```

Meaning

The output shows that the RIP-enabled interface on Device R1 is operational.

In general for this command, the output shows a list of the RIP neighbors that are configured on the device. Verify the following information:

- Each configured interface is present. Interfaces are listed in alphabetical order.
- Each configured interface is up. The state of the interface is listed in the **Local State** column. A state of **Up** indicates that the link is passing RIP traffic. A state of **Dn** indicates that the link is not passing RIP traffic. In a point-to-point link, this state generally means that either the end point is not configured for RIP or the link is unavailable.

Verifying the Exchange of RIP Messages

Purpose

Verify that RIP messages are being sent and received on all RIP-enabled interfaces.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R1> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0

fe-1/2/0.1:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                         Total   Last 5 min  Last minute
```

```
-------                    -----------  -----------  -----------
Updates Sent                      2669           10            2
Triggered Updates Sent               2            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received            2675           11            2
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures              0            0            0
RIP Requests Received                0            0            0
RIP Requests Ignored                 0            0            0
none                                 0            0            0
```

Meaning

The output shows the number of RIP routes learned. It also shows the number of RIP updates sent and received on the RIP-enabled interfaces. Verify the following information:

- The number of RIP routes learned matches the number of expected routes learned. Subnets learned by direct connectivity through an outgoing interface are not listed as RIP routes.
- RIP updates are being sent on each RIP-enabled interface. If no updates are being sent, the routing policy might not be configured to export routes.
- RIP updates are being received on each RIP-enabled interface. If no updates are being received, the routing policy might not be configured to export routes on the host connected to that subnet. The lack of updates might also indicate an authentication error.

Verifying Reachability of All Hosts in the RIP Network

Purpose

Use the `traceroute` command on each loopback address in the network to verify that all hosts in the RIP network are reachable from each Juniper Networks device.

Action

From operational mode, enter the `traceroute` command.

```
user@R1> traceroute 192.168.3.3
traceroute to 192.168.3.3 (192.168.3.3), 30 hops max, 40 byte packets
 1  10.0.0.2 (10.0.0.2)  1.094 ms  1.028 ms  0.957 ms
 2  192.168.3.3 (192.168.3.3)  1.344 ms  2.245 ms  2.125 ms
```

Meaning

Each numbered row in the output indicates a routing hop in the path to the host. The three-time increments indicate the round-trip time (RTT) between the device and the hop for each traceroute packet.

To ensure that the RIP network is healthy, verify the following information:

- The final hop in the list is the host you want to reach.
- The number of expected hops to the host matches the number of hops in the traceroute output. The appearance of more hops than expected in the output indicates that a network segment is probably unreachable. It might also indicate that the incoming or outgoing metric on one or more hosts has been set unexpectedly.

RIP Authentication

IN THIS SECTION

Understanding RIP Authentication

RIPv2 provides authentication support so that RIP links can require authentication keys (passwords) before they become active. Authentication provides an additional layer of security on the network beyond the other security features. By default, this authentication is disabled.

Authentication keys can be specified in either plain-text or MD5 form. Authentication requires all routers within the RIP network or subnetwork to have the same authentication type and key (password) configured.

This type of authentication is not supported on RIPv1 networks.

MD5 authentication uses an encoded MD5 checksum that is included in the transmitted packet. For MD5 authentication to work, both the receiving and transmitting routing devices must have the same MD5 key. You define an MD5 key for each interface. If MD5 is enabled on an interface, that interface accepts routing updates only if MD5 authentication succeeds. Otherwise, updates are rejected. The routing device only accepts RIPv2 packets sent using the same key identifier (ID) that is defined for that interface. Starting in Junos OS Release 20.3R1, we support multiple MD5 authentication keys for RIPv2 for increased security. This supports adding of MD5 keys with their `start-time`. RIPv2 packets are transmitted with MD5 authentication using the first configured key. RIPv2 authentication switches to the next key based on its configured key `start-time`. This provides automatic key switching without user intervention to change the MD5 keys as in the case of having only one MD5 key.

Note that the RIPv2 authentication described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

SEE ALSO

RIP Overview | 2

Enabling Authentication with Plain-Text Passwords

To configure authentication that requires a plain-text password to be included in the transmitted packet, enable simple authentication by performing these steps on all RIP devices in the network:

1. Navigate to the top of the configuration hierarchy.
2. Perform the configuration tasks described in Table 1 on page 25.
3. If you are finished configuring the router, commit the configuration.

Table 1: Configuring Simple RIP Authentication

Task	CLI Configuration Editor
Navigate to **Rip** level in the configuration hierarchy.	From the `[edit]` hierarchy level, enter **edit protocols rip**
Set the authentication type to **simple**.	Set the authentication type to **simple**: `set authentication-type simple`
Set the authentication key to a simple-text password. The password can be from 1 through 16 contiguous characters long and can include any ASCII strings.	Set the authentication key to a simple-text password: `set authentication-key` *password*

SEE ALSO

RIP Configuration Overview | 12

Example: Configuring Route Authentication for RIP using single MD5 key

IN THIS SECTION

This example shows how to configure authentication for a RIP network.

Requirements

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

Overview

IN THIS SECTION

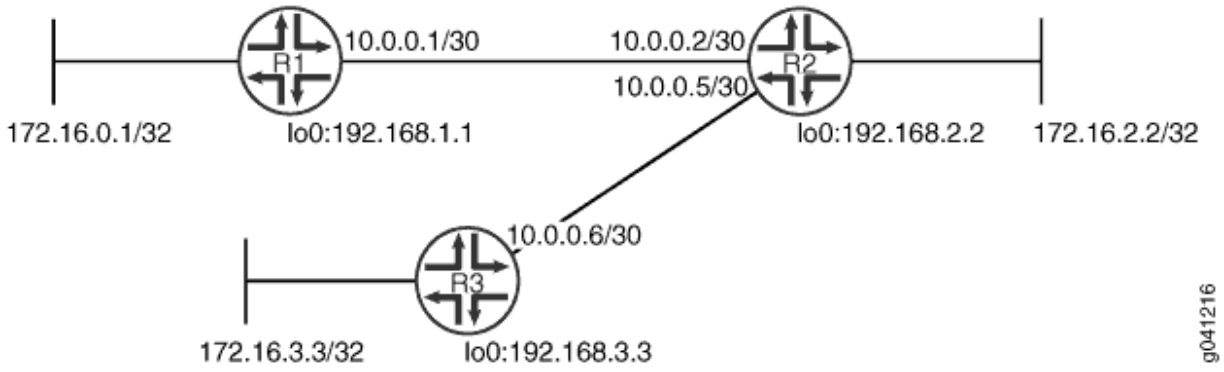
You can configure the router to authenticate RIP route queries. By default, authentication is disabled. You can use one of the following authentication methods:

- Simple authentication—Uses a text password that is included in the transmitted packet. The receiving router uses an authentication key (password) to verify the packet.
- MD5 authentication—Creates an encoded checksum that is included in the transmitted packet. The receiving router uses an authentication key (password) to verify the packet's MD5 checksum.

This example shows MD5 authentication.

Figure 6 on page 26 shows the topology used in this example.

Figure 6: RIP Authentication Network Topology



"CLI Quick Configuration" on page 27 shows the configuration for all of the devices in Figure 6 on page 26. The section "No Link Title" on page 28 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

- Procedure | 27

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
set protocols rip authentication-type md5
set protocols rip authentication-key "$ABC123$ABC123"
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
```

```
set protocols rip group rip-group neighbor fe-1/2/1.5
set protocols rip authentication-type md5
set protocols rip authentication-key "$ABC123$ABC123"
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set protocols rip authentication-type md5
set protocols rip authentication-key "$ABC123$ABC123"
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 RIP authentication:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
```

```
user@R1# set lo0 unit 1 family inet address 172.16.0.1/32
user@R1# set lo0 unit 1 family inet address 192.168.1.1/32
```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

```
[edit protocols rip group rip-group]
user@R1# set neighbor fe-1/2/0.1
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

5. Require MD5 authentication for RIP route queries received on an interface.

   The passwords must match on neighboring RIP routers. If the password does not match, the packet is rejected. The password can be from 1 through 16 contiguous characters long and can include any ASCII strings.

   Do not enter the password as shown here. The password shown here is the encrypted password that is displayed in the configuration after the actual password is already configured.

```
[edit protocols rip]
user@R1# set authentication-type md5
user@R1# set authentication-key "$ABC123$ABC123"
```

6. Configure tracing operations to track authentication.

```
[edit protocols rip traceoptions]
user@R1# set file rip-authentication-messages
user@R1# set flag auth
user@R1# set flag packets
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.0.1/32;
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
rip {
    traceoptions {
        file rip-authentication-messages;
        flag auth;
        flag packets;
    }
    authentication-type md5;
    authentication-key $ABC123$ABC123; ## SECRET-DATA
```

```
    group rip-group {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking for Authentication Failures

Purpose

Verify that there are no authentication failures.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R1> show rip statistics
RIPv2 info: port 520; holddown 120s.
```

```
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0

fe-1/2/0.1:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                     2669           10            2
Triggered Updates Sent              2            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
RIPv1 Updates Received              0            0            0
RIPv1 Bad Route Entries             0            0            0
RIPv1 Updates Ignored               0            0            0
RIPv2 Updates Received           2675           11            2
RIPv2 Bad Route Entries             0            0            0
RIPv2 Updates Ignored               0            0            0
Authentication Failures             0            0            0
RIP Requests Received               0            0            0
RIP Requests Ignored                0            0            0
none                                0            0            0
```

Meaning

The output shows that there are no authentication failures.

Verifying That MD5 Authentication Is Enabled in RIP Update Packets

Purpose

Use tracing operations to verify that MD5 authentication is enabled in RIP updates.

Action

From operational mode, enter the `show log` command.

```
user@R1> show log rip-authentication-messages | match md5
Feb 15 15:45:13.969462          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:45:43.229867          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:46:13.174410          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:46:42.716566          sending msg 0xb9a8c04, 3 rtes (needs MD5)
```

```
Feb 15 15:47:11.425076          sending msg 0xb9a8c04, 3 rtes (needs MD5)
...
```

Meaning

The **(needs MD5)** output shows that all route updates require MD5 authentication.

SEE ALSO

Understanding Basic RIP Routing | 13

Example: Configuring Route Authentication for RIP using multiple MD5 keys

IN THIS SECTION

This example shows how to configure authentication for a RIP network using multiple MD5 keys and how to configure a transition of MD5 keys on a RIP interface.

Requirements

This example uses the following hardware and software components:.

- Three ACX Series routers
- Junos OS Release 20.3 or later

Overview

MD5 authentication uses an encoded MD5 checksum that is included in the transmitted packet. For MD5 authentication to work, both the receiving and transmitting routing devices must have the same MD5 key.

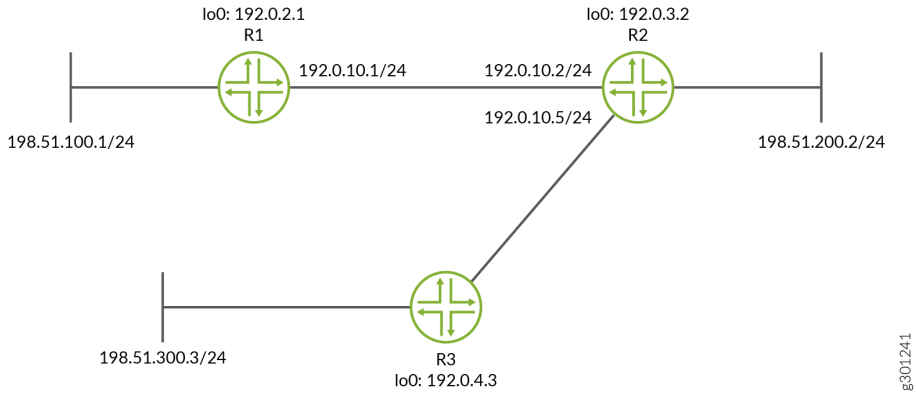
You define an MD5 key for each interface. If MD5 is enabled on an interface, that interface accepts routing updates only if MD5 authentication succeeds. Otherwise, updates are rejected. The routing device only accepts RIPv2 packets sent using the same key identifier (ID) that is defined for that interface.

For increased security, you can configure multiple MD5 keys, each with a unique key ID, and set the date and time to switch to a new key. The receiver of the RIPv2 packet uses the ID to determine which key to use for authentication. RIPv2 with multiple MD5 key feature supports adding of MD5 keys with their start-time. RIPv2 packets are transmitted with MD5 authentication using the first configured key. RIPv2 authentication switches to the next key based on its configured respective key start-time. This provides automatic key switching without user intervention to change the MD5 keys as in case of having only one MD5 key.

This example shows RIPv2 mutliple MD5 keys authentication.

Figure 7 on page 34 shows the topology used in this example.

Figure 7: Network Topology for RIP Authentication using multiple MD5 keys



"CLI Quick Configuration" on page 35 shows the configuration for all of the devices in Figure 7 on page 34. The section "CLI Quick Configuration" on page 35 describes the steps on Device R1.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
set protocols rip authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
set protocols rip authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01
set protocols rip authentication-selective-md5 4 key $XYZ123$XYZ123 start-time 2020-04-03.03:01
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
```

```
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set protocols rip authentication-type md5
set protocols rip authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
set protocols rip authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01
set protocols rip authentication-selective-md5 4 key $XYZ123$XYZ123 start-time 2020-04-03.03:01
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set protocols rip authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
set protocols rip authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01
set protocols rip authentication-selective-md5 4 key $XYZ123$XYZ123 start-time 2020-04-03.03:01
set protocols rip traceoptions file rip-authentication-messages
set protocols rip traceoptions flag auth
set protocols rip traceoptions flag packets
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 RIP authentication:

1. Configure the network interfaces.

This example shows multiple loopback interface addresses to simulate attached networks.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 192.0.10.1/24
user@R1# set lo0 unit 1 family inet address 198.51.100.1/24
user@R1# set lo0 unit 1 family inet address 192.0.2.1/32
```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

```
[edit protocols rip group rip-group]
user@R1# set neighbor fe-1/2/0
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

5. You can configure multiple MD5 keys by using different Key IDs. The key-IDs must match with the key-IDs of the neighboring RIP routers. If a router receives a packet with a key-id that is not within its configured set of keys, then the packet is rejected and is considered as authentication failure.

   The key-ID can be a number from 0 to 255 which uniquely identifies an MD5 key and the key value can be an ASCII string upto 16 characters long.

Do not enter the password as shown here. The password shown here is the encrypted password that is displayed in the configuration after the actual password is already configured.

```
[edit protocols rip]
user@R1# set authentication-selective-md5 key-id key key-value start-time time
user@R1# set authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
```

The `authentication-selective-md5` can be repeated to configure multiple keys.

6. If you want to migrate from an existing md5 authentication key, then you can configure another key with a start-time in future with enough leeway so as to allow configuring all the routers on the link. The transition to the new key is based on its start-time and it happens as soon as the clock reaches the start-time. You may delete keys that are no longer valid by entering the following command:.

```
[edit protocols rip]
user@host# delete authentication-selective-md5 key-id
```

NOTE: The start time is relevant for transmission only and not for receiving RIPv2 packets. Acceptance of received packets is based on the keys configured.

For example, if the time now is February 1, 2020, 1:00 AM and the following key is configured:

```
[edit protocols rip]
user@host# set authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
```

If you want to transition from this key to another key on March 2, at 2:00 AM, and you are able to configure all the routers on the link with the new key at the same time, then you may configure the following key:

```
[edit protocols rip]
user@host# set authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01
```

At 2:00 AM, once all the routers switch to the new key, you can safely delete key with id 2 by entering the following command.

```
[edit protocols rip]
user@host# delete authentication-selective-md5 2
```

7. Deletion of active key: If you delete the lastest active key, the system checks for the current configuration and uses the key with the latest key-ID within the existing configuration for RIPv2 packet transmission.

   For example, If you have configured the following keys with the key-ids:

```
[edit protocols rip]
user@R1# set authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01
user@R1#set authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01
user@R1#set authentication-selective-md5 4 key $XYZ123$XYZ123 start-time 2020-04-03.03:01
```

   The active key in this configuration is the key with key ID 4 and is used for sending the RIPv2 packet out. If you delete the active key ID 4, then the system checks for current configuration and looks for the key with the latest start-time, that is the key with ID 3 and uses it for packet transmission.

8. Configure tracing operations to track authentication.

```
[edit protocols rip traceoptions]
user@R1# set file rip-authentication-messages
user@R1# set flag auth
user@R1# set flag packets
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 192.0.10.1/24;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 198.51.100.1/24;
```

```
            address 192.0.2.1/32;
        }
    }
}
```

```
user@R1# show protocols
rip {
    traceoptions {
        file rip-authentication-messages;
        flag auth;
        flag packets;
    }
    authentication-selective-md5 2 key $ABC123$ABC123 start-time 2020-02-01.01:01 ## SECRET-DATA
    authentication-selective-md5 3 key $MNO123$MNO123 start-time 2020-03-02.02:01 ## SECRET-DATA
    authentication-selective-md5 4 key $XYZ123$XYZ123 start-time 2020-04-03.03:01 ## SECRET-DATA
    group rip-group {
        export advertise-routes-through-rip;
        neighbor ge-0/0/5.0;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking for Authentication Failures

Purpose

To check for authentication failures counters.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R1> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0

ge-0/0/5.0:  5 routes learned; 28 routes advertised; timeout 180s; update interval 30s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                     53058           20            4
Triggered Updates Sent               2            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received           26538           10            2
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures          23853            0            0
RIP Requests Received                0            0            0
RIP Requests Ignored                 0            0            0
none                                 0            0            0
```

Meaning

The `Authentication Failures` counter displays the authentication failures count. This output shows that the authentication failure count is 23853.

Checking for the current active MD5 key.

Purpose

To check for the current active key being used.

Action

From operational mode, enter the `show rip neighbor fe-1/2/0` command.

```
user@R1> show rip neighbor fe-1/2/0
                 Local  Source          Destination     Send   Receive   In
Neighbor         State  Address         Address         Mode   Mode     Met
--------         -----  -------         -----------     ----   -------  ---
fe-1/2/0          Up 14.14.14.1      224.0.0.9       mcast  both       1
  Auth type: SELECTIVE-MD5, Active key ID: 2, Start time: 1970 Jan  1 05:30:00 IST
```

Verifying That MD5 Authentication Is Enabled in RIP Update Packets

Purpose

Use tracing operations to verify that MD5 authentication is enabled in RIP updates.

Action

From operational mode, enter the `show log` command.

```
user@R1> show log rip-authentication-messages | match md5
Feb 15 15:45:13.969462          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:45:43.229867          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:46:13.174410          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:46:42.716566          sending msg 0xb9a8c04, 3 rtes (needs MD5)
Feb 15 15:47:11.425076          sending msg 0xb9a8c04, 3 rtes (needs MD5)
...
```

Meaning

The **(needs MD5)** output shows that all route updates require MD5 authentication.

SEE ALSO

Understanding Basic RIP Routing | 13

Change History Table

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

Release	Description
20.3R1	Starting in Junos OS Release 20.3R1, we support multiple MD5 authentication keys for RIPv2 for increased security
15.1X49	Note that the RIPv2 authentication described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

RIP Timers

IN THIS SECTION

Understanding RIP Timers

RIP uses several timers to regulate its operation.

The update interval is the interval at which routes that are learned by RIP are advertised to neighbors. This timer controls the interval between routing updates. The update interval is set to 30 seconds, by default, with a small random amount of time added when the timer is reset. This added time prevents congestion that can occur if all routing devices update their neighbors simultaneously.

To configure the update time interval, include the `update-interval` statement:

```
update-interval seconds;
```

seconds can be a value from 10 through 60.

You can set a route timeout interval. If a route is not refreshed after being installed in the routing table by the specified time interval, the route is marked as invalid and is removed from the routing table after the hold-down period expires.

To configure the route timeout for RIP, include the `route-timeout` statement:

```
route-timeout seconds;
```

seconds can be a value from 30 through 360. The default value is 180 seconds.

RIP routes expire when either a route timeout limit is met or a route metric reaches infinity, and the route is no longer valid. However, the expired route is retained in the routing table for a specified period so that neighbors can be notified that the route has been dropped. This time period is set by configuring the hold-down timer. Upon expiration of the hold-down timer, the route is removed from the routing table.

To configure the hold-down timer for RIP, include the `holddown` statement:

```
holddown seconds;
```

seconds can be a value from 10 through 180. The default value is 120 seconds.

NOTE: In Junos OS Release 11.1 and later, a retransmission timer is available for RIP demand circuits.

Generally, we recommend against changing the RIP timers, unless the effects of a change are well understood. The route timeout should be at least three times the update interval. Normally, the default values are best left in effect for standard operations.

Example: Configuring RIP Timers

IN THIS SECTION

This example shows how to configure the RIP update interval and how to monitor the impact of the change.

Requirements

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

Overview

IN THIS SECTION

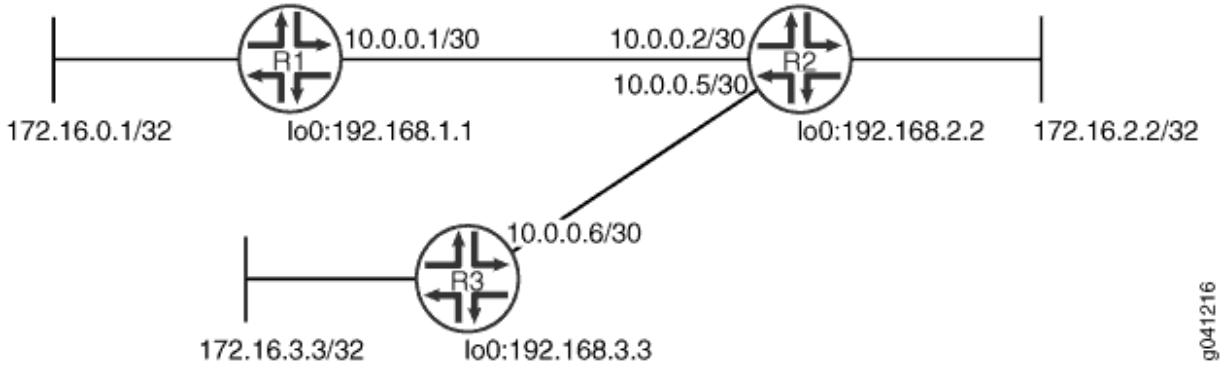
In this example, Device R2 has an update interval of 60 seconds for its neighbor, Device R1, and an update interval of 10 seconds for its neighbor, Device R3.

This example is not necessarily practical, but it is shown for demonstration purposes. Generally, we recommend against changing the RIP timers, unless the effects of a change are well understood. Normally, the default values are best left in effect for standard operations.

An export policy is also shown because an export policy is required as part of the minimum configuration for RIP.

Figure 8 on page 46 shows the topology used in this example.

Figure 8: RIP Timers Network Topology



"CLI Quick Configuration" on page 46 shows the configuration for all of the devices in Figure 8 on page 46. The section "No Link Title" on page 47 describes the steps on Device R2.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
```

```
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2 update-interval 60
set protocols rip group rip-group neighbor fe-1/2/1.5 update-interval 10
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Step-by-Step Procedure

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

To configure the RIP update interval:

1. Configure the network interfaces.

This example shows multiple loopback interface addresses to simulate attached networks.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 2 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 5 family inet address 10.0.0.5/30
user@R2# set lo0 unit 2 family inet address 192.168.2.2/32
user@R2# set lo0 unit 2 family inet address 172.16.2.2/32
```

2. Configure different update intervals for the two RIP neighbors.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

```
[edit protocols rip group rip-group]
user@R2# set neighbor fe-1/2/0.2 update-interval 60
user@R2# set neighbor fe-1/2/1.5 update-interval 10
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R2# set from protocol direct
user@R2# set from protocol rip
user@R2# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R2# set export advertise-routes-through-rip
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 5 {
        family inet {
            address 10.0.0.5/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.168.2.2/32;
            address 172.16.2.2/32;
        }
    }
}
```

```
user@R2# show protocols
rip {
    group rip-group {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.2 {
            update-interval 60;
        }
        neighbor fe-1/2/1.5 {
            update-interval 10;
        }
```

```
    }
}
```

```
user@R2# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the RIP Updates Sent by Device R2

Purpose

Make sure that the RIP update packets are sent at the expected interval.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R2> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              4              2              0              0
```

```
fe-1/2/0.2:  2 routes learned; 5 routes advertised; timeout 180s; update interval 60s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                      123            5            1
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
RIPv1 Updates Received              0            0            0
RIPv1 Bad Route Entries             0            0            0
RIPv1 Updates Ignored               0            0            0
RIPv2 Updates Received            244           10            2
RIPv2 Bad Route Entries             0            0            0
RIPv2 Updates Ignored               0            0            0
Authentication Failures             0            0            0
RIP Requests Received               0            0            0
RIP Requests Ignored                0            0            0
none                                0            0            0

fe-1/2/1.5:  2 routes learned; 5 routes advertised; timeout 180s; update interval 10s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                      734           32            6
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
RIPv1 Updates Received              0            0            0
RIPv1 Bad Route Entries             0            0            0
RIPv1 Updates Ignored               0            0            0
RIPv2 Updates Received            245           11            2
RIPv2 Bad Route Entries             0            0            0
RIPv2 Updates Ignored               0            0            0
Authentication Failures             0            0            0
RIP Requests Received               0            0            0
RIP Requests Ignored                0            0            0
none                                0            0            0
```

Meaning

The **update interval** field shows that the interval is 60 seconds for Neighbor R1 and 10 seconds for Neighbor R3. The **Updates Sent** field shows that Device R2 is sending updates to Device R1 at roughly 1/6 of the rate that it is sending updates to Device R3.

Checking the RIP Updates Received by Device R2

Purpose

Make sure that the RIP update packets are sent at the expected interval.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R1> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0

fe-1/2/0.1:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                       312           10            2
Triggered Updates Sent               2            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received             181            5            1
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures              0            0            0
RIP Requests Received                1            0            0
RIP Requests Ignored                 0            0            0
none                                 0            0            0
```

Meaning

The **RIPv2 Updates Received** field shows the number of updates received from Device R2.

Checking the RIP Updates Received by Device R3

Purpose

Make sure that the RIP update packets are sent at the expected interval.

Action

From operational mode, enter the `show rip statistics` command.

```
user@R3> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0

fe-1/2/0.6:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                       314           11            2
Triggered Updates Sent               1            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received             827           31            6
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures              0            0            0
RIP Requests Received                0            0            0
RIP Requests Ignored                 0            0            0
none                                 0            0            0
```

Meaning

The **RIPv2 Updates Received** field shows the number of updates received from Device R2.

Change History Table

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

Release	Description
11.1	In Junos OS Release 11.1 and later, a retransmission timer is available for RIP demand circuits.

RIP Demand Circuits

IN THIS SECTION

RIP Demand Circuits Overview

IN THIS SECTION

RIP periodically sends routing information (RIP packets) to neighboring devices. These periodic broadcasts can consume bandwidth resources and interfere with network traffic by preventing WAN circuits from being closed. Demand circuits for RIP is defined in RFC 2091 and overcomes these issues by exchanging incremental updates on demand.

A demand circuit is a point-to-point connection between two neighboring interfaces configured for RIP. Demand circuits preserve bandwidth by establishing a link when data needs to be transferred, and terminating the link when the data transfer is complete. Demand circuits increase the efficiency of RIP on the configured interfaces by offering minimal network overhead in terms of messages passed between the demand circuit end points, conserving resources, and reducing costs.

By configuring RIP demand circuits, a specific event triggers the device to send an update, thereby eliminating the periodic transmission of RIP packets over the neighboring interface. To save overhead, the device sends RIP information only when changes occur in the routing database, such as:

- The device is first powered on
- The device receives a request for route update information
- A change occurs in the network
- The demand circuit goes down or comes up

The device sends update requests, update responses, and acknowledgments. In addition, the device retransmits updates and requests until valid acknowledgments are received. The device dynamically learns RIP neighbors. If the neighboring interface goes down, RIP flushes routes learned from the neighbor's IP address.

Routes learned from demand circuits do not age like other RIP entries because demand circuits are in a permanent state. Routes in a permanent state are only removed under the following conditions:

- A formerly reachable route changes to unreachable in an incoming response
- The demand circuit is down due to an excessive number of unacknowledged retransmissions

You can also set the *RIP hold-down timer* and the RIP demand circuit retransmission timer to regulate performance. The demand circuit uses these timers to determine if there is a change that requires update messages to be sent. There is also a database timer that runs only when RIP flushes learned routes from the routing table.

This topic includes the following sections:

- "RIP Demand Circuit Packets" on page 55
- "Timers Used by RIP Demand Circuits " on page 56

RIP Demand Circuit Packets

When you configure an interface for RIP demand circuits, the supported command field packet types are different than those for RIP version 1 and RIP version 2. RIP packets for RIP demand circuits contain three additional packet types and an extended 4-byte update header. Both RIP version 1 and RIP version 2 support the three packet types and the extended 4-byte header. Table 2 on page 56 describes the three packet types.

Table 2: RIP Demand Circuit Packet Types

Packet Type	Description
Update Request	Update request messages seek information for the device's routing table. This message is sent when the device is first powered on or when a down demand circuit comes up. The device sends this message every 5 seconds (by default) until an update response message is received.
Update Response	Update response messages are sent in response to an update request message, which occurs when the device is first powered on or when a down demand circuit comes up. Each update response message contains a sequence number that the neighbor uses to acknowledge the update request.
Update Acknowledge	Update acknowledge messages are sent in response to every update response message received by the neighbor.

NOTE: These packets are only valid on interfaces configured for RIP demand circuits. If a demand circuit receives a RIP packet that does not contain these packet types, it silently discards the packet and logs an error message similar to the following:

Ignoring RIP packet with invalid version 0 from neighbor 10.0.0.0 and source 10.0.0.1

Timers Used by RIP Demand Circuits

RIP demand circuits use the RIP hold-down timer and the RIP demand circuit retransmission timer to regulate performance and to determine if there is a change in the network that requires the device to send update messages. The hold-down timer is a global RIP timer that affects the entire RIP configuration; whatever range you configure for RIP applies to RIP demand circuits. The retransmission timer affects only RIP demand circuits. In addition, there is a database timer that runs only when RIP flushes learned routes from the routing table.

- Hold-down timer (global RIP timer)—Use the hold-down timer to configure the number of seconds that RIP waits before updating the routing table. The value of the hold-down timer affects the entire RIP configuration, not just the demand circuit interfaces. The hold-down timer starts when a route timeout limit is met, when a formerly reachable route is unreachable, or when a demand circuit interface is down. When the hold-down timer is running, routes are advertised as unreachable on other interfaces. When the hold-down timer expires, the route is removed from the routing table if all destinations are aware that the route is unreachable or the remaining destinations are down. By default, RIP waits 120 seconds between routing table updates. The range is from 10 to 180 seconds.

- Retransmission timer (RIP demand circuit timer)—RIP demand circuits send update messages every 5 seconds to an unresponsive peer. Use the retransmission timer to limit the number of times a demand circuit resends update messages to an unresponsive peer. If the configured retransmission threshold is reached, routes from the next hop router are marked as unreachable and the hold-down timer starts. The value of the retransmission timer affects only the demand circuit interfaces. To determine the number of times to resend the update message, use the following calculation:

  ```
  5 seconds * number of retransmissions = retransmission seconds
  ```

  The retransmission range is from 5 through 180 seconds, which corresponds to sending an update message a minimum of 1 time (5 seconds) and a maximum of 36 times (180 seconds).

- Database timer (global timeout timer)—Routes learned from demand circuits do not age like other RIP entries because demand circuits are in a permanent state. On a RIP demand circuit, the database timer starts upon receipt of the update response message with the flush flag sent from a RIP demand circuit peer. When the neighbor receives this message, all routes from that peer are flushed, and the database timer starts and runs for the configured route timeout interval. When the database timer is running, routes are still advertised as reachable on other interfaces. When the database timer expires, the device advertises all routes from its peer as unreachable.

SEE ALSO

RIP Overview | 2

demand-circuit

Example: Configuring RIP Timers | 44

Example: Configuring RIP Demand Circuits | 57

holddown

max-retrans-time

Example: Configuring RIP Demand Circuits

IN THIS SECTION

This example describes how to configure an interface as a RIP demand circuit.

Requirements

Before you begin, configure the device interfaces. See the Junos OS Network Interfaces Library for Routing Devices or the *Junos OS Interfaces Configuration Guide for Security Devices*.

Overview

A demand circuit is a point-to-point connection between two neighboring interfaces configured for RIP. Demand circuits increase the efficiency of RIP on the configured interfaces by eliminating the periodic transmission of RIP packets. Demand circuits preserve bandwidth by establishing a link when data needs to be transferred, and terminating the link when the data transfer is complete. In this example, two devices are connected using SONET/SDH interfaces.

NOTE: When you configure RIP demand circuits, any silent removal of the RIP configuration goes unnoticed by the RIP peer and leads to stale entries in the routing table. To clear the stale entries, deactivate and reactivate RIP on the neighboring devices.

In this example, you configure interface **so-0/1/0** with the following settings:

- **demand-circuit**—Configures the interface as a demand circuit. To complete the demand circuit, you must configure both ends of the pair as demand circuits.
- **max-retrans-time**—RIP demand circuits send update messages every 5 seconds to an unresponsive peer. Use the retransmission timer to limit the number of times a demand circuit resends update messages to an unresponsive peer. If the configured retransmission threshold is reached, routes from the next-hop router are marked as unreachable, and the hold-down timer starts. The value of the retransmission timer affects only the demand circuit interfaces. To determine the number of times to resend the update message, use the following calculation:

```
5 seconds x retransmissions = retransmission seconds
```

For example, if you want the demand circuit to send only two update messages to an unresponsive peer, the calculation is: 5 x 2 = 10. When you configure the retransmission timer, you enter 10 seconds.

The retransmission range is from 5 through 180 seconds, which corresponds to sending an update message a minimum of 1 time (5 seconds) and a maximum of 36 times (180 seconds).

Configuration

IN THIS SECTION

In the following example, you configure a neighboring interface to be a RIP demand circuit and save the configuration.

Procedure

CLI Quick Configuration

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

```
set interfaces so-0/1/0 unit 0 family inet address 192.0.2.0/24
set protocols rip group group1 neighbor so-0/1/0 demand-circuit
set protocols rip group group1 neighbor so-0/1/0 max-retrans-time 10
```

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 RIP demand circuit on one neighboring interface:

1. Configure the interface.

```
[edit interfaces]
user@host# set so-0/1/0 unit 0 family inet address 192.0.2.0/24
```

2. Configure the neighbor as a demand circuit.

```
[edit protocols rip]
user@host# set group group1 neighbor so-0/1/0 demand-circuit
```

3. Configure the demand circuit retransmission timer.

```
[edit protocols rip]
user@host# set group group1 neighbor so-0/1/0 max-retrans-time 10
```

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

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

NOTE: Repeat this entire configuration on the other neighboring interface.

Results

Confirm your configuration by entering the `show interfaces` and `show protocols` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show interfaces
so-0/1/0 {
    unit 0 {
        family inet {
            address 192.0.2.0/24;
        }
```

```
    }
}
```

```
user@host# show protocols
rip {
    group group1 {
        neighbor so-0/1/0 {
             demand-circuit;
             max-retrans-time 10;
        }
    }
}
```

Verification

IN THIS SECTION

- Verifying a Demand Circuit Configuration | **61**

Confirm that the configuration is working properly.

Verifying a Demand Circuit Configuration

Purpose

Verify that the demand circuit configuration is working.

Action

To verify that the demand circuit configuration is in effect, use the **show rip neighbor** operational mode command.

```
user@host> show rip neighbor
                         Source          Destination     Send   Receive  In
Neighbor          State  Address         Address         Mode   Mode     Met
```

```
--------          -----  -------        -----------     ----   -------  ---
so-0/1/0.0(DC)       Up  10.10.10.2     224.0.0.9       mcast  both       1
```

When you configure demand circuits, the `show rip neighbor` command displays a DC flag next to the neighboring interface configured for demand circuits.

NOTE: If you configure demand circuits at the [`edit protocols rip group` *`group-name`* `neighbor` *`neighbor-name`*] hierarchy level, the output shows only the neighboring interface that you specifically configured as a demand circuit. If you configure demand circuits at the [`edit protocols rip group` *`group-name`*] hierarchy level, all of the interfaces in the group are configured as demand circuits. Therefore, the output shows all of the interfaces in that group as demand circuits.

BFD for RIP

IN THIS SECTION

Understanding BFD for RIP

The Bidirectional Forwarding Detection (BFD) Protocol is a simple hello mechanism that detects failures in a network. Hello packets are sent at a specified, regular interval. A neighbor failure is detected when the routing device stops receiving a reply after a specified interval. BFD works with a wide variety of network environments and topologies. BFD failure detection times are shorter than RIP detection times, providing faster reaction times to various kinds of failures in the network. Instead of waiting for the routing protocol neighbor timeout, BFD provides rapid detection of link failures. BFD timers are adaptive and can be adjusted to be more or less aggressive. For example, a timer can adapt to a higher value if the adjacency fails, or a neighbor can negotiate a higher value for a timer than the one

configured. Note that the functionality of configuring BFD for RIP described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

NOTE: EX4600 and QFX5000 Series switches running Junos OS or Junos OS Evolved do not support minimum interval values of less than 1 second in centralized and distributed mode.

BFD enables quick failover between a primary and a secondary routed path. The protocol tests the operational status of the interface multiple times per second. BFD provides for configuration timers and thresholds for failure detection. For example, if the minimum interval is set for 50 milliseconds and the threshold uses the default value of three missed messages, a failure is detected on an interface within 200 milliseconds of the failure.

Intervening devices (for example, an Ethernet LAN switch) hide link-layer failures from routing protocol peers, such as when two routers are connected by way of a LAN switch, where the local interface status remains up even when a physical fault happens on the remote link. Link-layer failure detection times vary, depending on the physical media and the Layer 2 encapsulation. BFD can provide fast failure detection times for all media types, encapsulations, topologies, and routing protocols.

To enable BFD for RIP, both sides of the connection must receive an update message from the peer. By default, RIP does not export any routes. Therefore, you must enable update messages to be sent by configuring an export policy for routes before a BFD session is triggered.

Example: Configuring BFD for RIP

IN THIS SECTION

This example shows how to configure Bidirectional Forwarding Detection (BFD) for a RIP network.

Requirements

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

Overview

IN THIS SECTION

To enable failure detection, include the `bfd-liveness-detection` statement:

```
bfd-liveness-detection {
    detection-time {
        threshold milliseconds;
    }
    minimum-interval milliseconds;
    minimum-receive-interval milliseconds;
    multiplier number;
    no-adaptation;
    transmit-interval {
        threshold milliseconds;
        minimum-interval milliseconds;
    }
    version (1 | automatic);
}
```

Optionally, you can specify the threshold for the adaptation of the detection time by including the `threshold` statement. When the BFD session detection time adapts to a value equal to or greater than the threshold, a single trap and a system log message are sent.

To specify the minimum transmit and receive interval for failure detection, include the `minimum-interval` statement. This value represents the minimum interval at which the local routing device transmits hello packets as well as the minimum interval at which the routing device expects to receive a reply from a neighbor with which it has established a BFD session. You can configure a value in the range from 1 through 255,000 milliseconds. This examples sets a minimum interval of 600 milliseconds.

NOTE: Depending on your network environment, these additional recommendations might apply:

- The reccomended minimum interval for distributed BFD is 100 ms with a `multiplier` of 3.
- For very large-scale network deployments with a large number of BFD sessions, contact Juniper Networks customer support for more information.
- For BFD sessions to remain up during a Routing Engine switchover event when nonstop active routing (NSR) is configured, specify a minimum interval of 2500 ms for Routing Engine-based sessions. For distributed BFD sessions with nonstop active routing configured, the minimum interval recommendations are unchanged and depend only on your network deployment.

You can optionally specify the minimum transmit and receive intervals separately.

To specify only the minimum receive interval for failure detection, include the `minimum-receive-interval` statement. This value represents the minimum interval at which the local routing device expects to receive a reply from a neighbor with which it has established a BFD session. You can configure a value in the range from 1 through 255,00 milliseconds.

To specify only the minimum transmit interval for failure detection, include the `transmit-interval minimum-interval` statement. This value represents the minimum interval at which the local routing device transmits hello packets to the neighbor with which it has established a BFD session. You can configure a value in the range from 1 through 255,000 milliseconds.

To specify the number of hello packets not received by a neighbor that causes the originating interface to be declared down, include the `multiplier` statement. The default is 3, and you can configure a value in the range from 1 through 255.

To specify the threshold for detecting the adaptation of the transmit interval, include the `transmit-interval threshold` statement. The threshold value must be greater than the transmit interval.

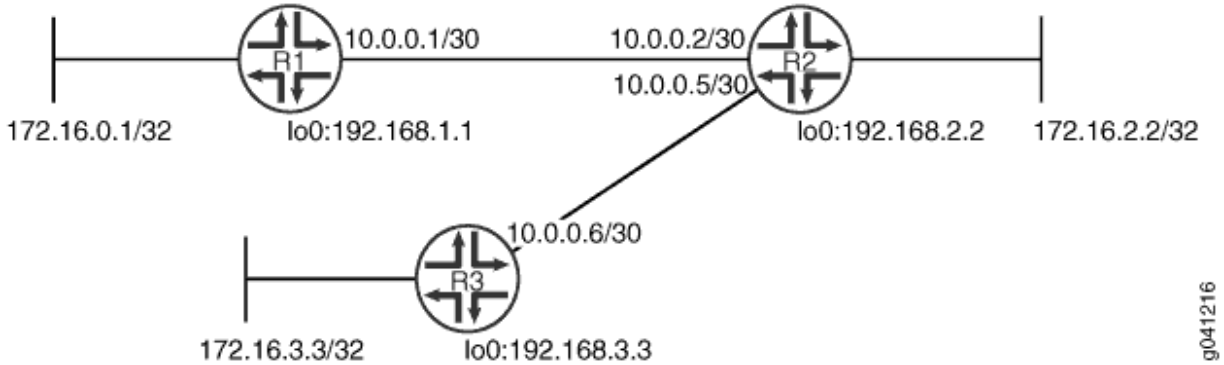
To specify the BFD version used for detection, include the `version` statement. The default is to have the version detected automatically.

You can trace BFD operations by including the `traceoptions` statement at the `[edit protocols bfd]` hierarchy level.

In Junos OS Release 9.0 and later, you can configure BFD sessions not to adapt to changing network conditions. To disable BFD adaptation, include the `no-adaptation` statement. We recommend that you not disable BFD adaptation unless it is preferable not to have BFD adaptation enabled in your network.

Figure 9 on page 66 shows the topology used in this example.

Figure 9: RIP BFD Network Topology



"CLI Quick Configuration" on page 66 shows the configuration for all of the devices in Figure 9 on page 66. The section "Step-by-Step Procedure" on page 67 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set protocols bfd traceoptions file bfd-trace
set protocols bfd traceoptions flag all
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
```

```
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 a BFD for a RIP network:

1. Configure the network interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

```
[edit protocols rip group rip-group]
user@R1# set neighbor fe-1/2/0.1
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

5. Enable BFD.

```
[edit protocols rip group rip-group]
user@R1# set bfd-liveness-detection minimum-interval 600
```

6. Configure tracing operations to track BFD messages.

```
[edit protocols bfd traceoptions]
user@R1# set file bfd-trace
user@R1# set flag all
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
```

```
user@R1# show protocols
bfd {
    traceoptions {
        file bfd-trace;
        flag all;
    }
}
rip {
    group rip-group {
        export advertise-routes-through-rip;
        bfd-liveness-detection {
            minimum-interval 600;
        }
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying That the BFD Sessions Are Up

Purpose

Make sure that the BFD sessions are operating.

Action

From operational mode, enter the `show bfd session` command.

```
user@R1> show bfd session
                                                Detect   Transmit
Address                  State     Interface    Time     Interval  Multiplier
10.0.0.2                 Up        fe-1/2/0.1   1.800     0.600        3

1 sessions, 1 clients
Cumulative transmit rate 1.7 pps, cumulative receive rate 1.7 pps
```

Meaning

The output shows that there are no authentication failures.

Checking the BFD Trace File

Purpose

Use tracing operations to verify that BFD packets are being exchanged.

Action

From operational mode, enter the `show log` command.

```
user@R1> show log bfd-trace
Feb 16 10:26:32 PPM Trace: BFD periodic xmit to 10.0.0.2 (IFL 124, rtbl 53, single-hop port)
Feb 16 10:26:32 Received Downstream TraceMsg (24) len 86:
Feb 16 10:26:32    IfIndex (3) len 4: 0
Feb 16 10:26:32    Protocol (1) len 1: BFD
Feb 16 10:26:32    Data (9) len 61: (hex) 42 46 44 20 70 61 63 6b 65 74 20 66 72 6f 6d 20 31 30
2e
Feb 16 10:26:32 PPM Trace: BFD packet from 10.0.0.1 (IFL 73, rtbl 56, ttl 255) absorbed
Feb 16 10:26:32 Received Downstream TraceMsg (24) len 60:
Feb 16 10:26:32    IfIndex (3) len 4: 0
Feb 16 10:26:32    Protocol (1) len 1: BFD
Feb 16 10:26:32    Data (9) len 35: (hex) 42 46 44 20 70 65 72 69 6f 64 69 63 20 78 6d 69 74 20
6f
...
```

Meaning

The output shows the normal functioning of BFD.

Understanding BFD Authentication for RIP

IN THIS SECTION

BFD enables rapid detection of communication failures between adjacent systems. By default, authentication for BFD sessions is disabled. However, when running BFD over Network Layer protocols, the risk of service attacks can be significant. We strongly recommend using authentication if you are running BFD over multiple hops or through insecure tunnels. Beginning with Junos OS Release 9.6, Junos OS supports authentication for BFD sessions running over RIP. BFD authentication is only supported in the domestic image and is not available in the export image.

You authenticate BFD sessions by specifying an authentication algorithm and keychain, and then associating that configuration information with a security authentication keychain using the keychain name.

The following sections describe the supported authentication algorithms, security keychains, and the level of authentication that can be configured:

BFD Authentication Algorithms

Junos OS supports the following algorithms for BFD authentication:

- **simple-password**—Plain-text password. One to 16 bytes of plain text are used to authenticate the BFD session. One or more passwords can be configured. This method is the least secure and should be used only when BFD sessions are not subject to packet interception.
- **keyed-md5**—Keyed Message Digest 5 hash algorithm for sessions with transmit and receive intervals greater than 100 ms. To authenticate the BFD session, keyed MD5 uses one or more secret keys (generated by the algorithm) and a sequence number that is updated periodically. With this method, packets are accepted at the receiving end of the session if one of the keys matches and the sequence number is greater than or equal to the last sequence number received. Although more secure than a simple password, this method is vulnerable to replay attacks. Increasing the rate at which the sequence number is updated can reduce this risk.
- **meticulous-keyed-md5**—Meticulous keyed Message Digest 5 hash algorithm. This method works in the same manner as keyed MD5, but the sequence number is updated with every packet. Although more secure than keyed MD5 and simple passwords, this method might take additional time to authenticate the session.
- **keyed-sha-1**—Keyed Secure Hash Algorithm I for sessions with transmit and receive intervals greater than 100 ms. To authenticate the BFD session, keyed SHA uses one or more secret keys (generated by the algorithm) and a sequence number that is updated periodically. The key is not carried within the packets. With this method, packets are accepted at the receiving end of the session if one of the keys matches and the sequence number is greater than the last sequence number received.
- **meticulous-keyed-sha-1**—Meticulous keyed Secure Hash Algorithm I. This method works in the same manner as keyed SHA, but the sequence number is updated with every packet. Although more secure than keyed SHA and simple passwords, this method might take additional time to authenticate the session.

NOTE: *Nonstop active routing* is not supported with meticulous-keyed-md5 and meticulous-keyed-sha-1 authentication algorithms. BFD sessions using these algorithms might go down after a switchover.

NOTE: QFX5000 Series switches and EX4600 switches do not support minimum interval values of less than 1 second.

Security Authentication Keychains

The security authentication keychain defines the authentication attributes used for authentication key updates. When the security authentication keychain is configured and associated with a protocol through the keychain name, authentication key updates can occur without interrupting routing and signaling protocols.

The authentication keychain contains one or more keychains. Each keychain contains one or more keys. Each key holds the secret data and the time at which the key becomes valid. The algorithm and keychain must be configured on both ends of the BFD session, and they must match. Any mismatch in configuration prevents the BFD session from being created.

BFD allows multiple clients per session, and each client can have its own keychain and algorithm defined. To avoid confusion, we recommend specifying only one security authentication keychain.

Strict Versus Loose Authentication

By default, strict authentication is enabled and authentication is checked at both ends of each BFD session. Optionally, to smooth migration from nonauthenticated sessions to authenticated sessions, you can configure *loose checking*. When loose checking is configured, packets are accepted without authentication being checked at each end of the session. This feature is intended for transitional periods only.

SEE ALSO

bfd-liveness-detection

Example: Configuring BFD Authentication for RIP

IN THIS SECTION

This example shows how to configure Bidirectional Forwarding Detection (BFD) authentication for a RIP network.

Requirements

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

The devices must be running Junos OS Release 9.6 or later.

Overview

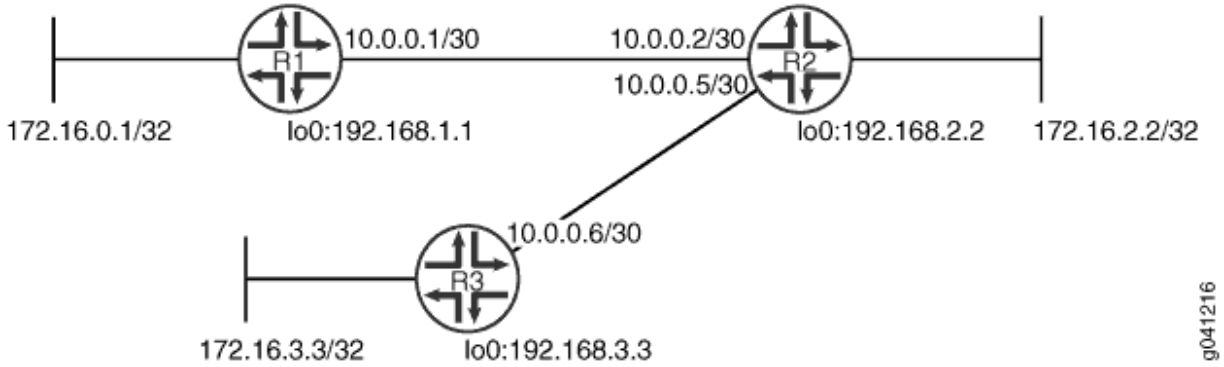
IN THIS SECTION

Only three steps are needed to configure authentication on a BFD session:

1. Specify the BFD authentication algorithm for the RIP protocol.
2. Associate the authentication keychain with the RIP protocol.
3. Configure the related security authentication keychain.

Figure 10 on page 75 shows the topology used in this example.

Figure 10: RIP BFD Authentication Network Topology



"CLI Quick Configuration" on page 75 shows the configuration for all of the devices in Figure 10 on page 75. The section "No Link Title" on page 77 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

- Procedure | 75

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set protocols bfd traceoptions file bfd-trace
set protocols bfd traceoptions flag all
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
```

```
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set protocols rip group rip-group bfd-liveness-detection authentication key-chain bfd-rip
set protocols rip group rip-group bfd-liveness-detection authentication algorithm keyed-md5
set protocols rip group rip-group bfd-liveness-detection authentication loose-check
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
set security authentication-key-chains key-chain bfd-rip key 53 secret $ABC123$ABC123
set security authentication-key-chains key-chain bfd-rip key 53 start-time "2012-2-16.12:00:00
-0800"
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set protocols rip group rip-group bfd-liveness-detection authentication key-chain bfd-rip
set protocols rip group rip-group bfd-liveness-detection authentication algorithm keyed-md5
set protocols rip group rip-group bfd-liveness-detection authentication loose-check
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
set security authentication-key-chains key-chain bfd-rip key 53 secret $ABC123$ABC123
set security authentication-key-chains key-chain bfd-rip key 53 start-time "2012-2-16.12:00:00
-0800"
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set protocols rip group rip-group bfd-liveness-detection minimum-interval 600
set protocols rip group rip-group bfd-liveness-detection authentication key-chain bfd-rip
set protocols rip group rip-group bfd-liveness-detection authentication algorithm keyed-md5
set protocols rip group rip-group bfd-liveness-detection authentication loose-check
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

```
set security authentication-key-chains key-chain bfd-rip key 53 secret $ABC123$ABC123
set security authentication-key-chains key-chain bfd-rip key 53 start-time "2012-2-16.12:00:00
-0800"
```

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 a BFD authentication:

1. Configure the network interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

```
[edit protocols rip group rip-group]
user@R1# set neighbor fe-1/2/0.1
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

5. Enable BFD.

```
[edit protocols rip group rip-group]
user@R1# set bfd-liveness-detection minimum-interval 600
```

6. Specify the algorithm (**keyed-md5**, **keyed-sha-1**, **meticulous-keyed-md5**, **meticulous-keyed-sha-1**, or **simple-password**) to use.

NOTE: Nonstop active routing is not supported with meticulous-keyed-md5 and meticulous-keyed-sha-1 authentication algorithms. BFD sessions using these algorithms might go down after a switchover.

```
[edit protocols rip group rip-group]
user@R1# set bfd-liveness-detection authentication algorithm keyed-md5
```

7. Specify the keychain to be used to associate BFD sessions on RIP with the unique security authentication keychain attributes.

   The keychain you specify must match a keychain name configured at the `[edit security authentication key-chains]` hierarchy level.

   The algorithm and keychain must be configured on both ends of the BFD session, and they must match. Any mismatch in configuration prevents the BFD session from being created.

```
[edit protocols rip group rip-group]
user@R1# set bfd-liveness-detection authentication key-chain bfd-rip
```

8. (Optional) Specify loose authentication checking if you are transitioning from nonauthenticated sessions to authenticated sessions.

```
[edit protocols rip group rip-group]
user@R1# set bfd-liveness-detection authentication loose-check
```

9. Specify the unique security authentication information for BFD sessions:

   - The matching keychain name as specified in Step "7" on page 78.
   - At least one key, a unique integer between **0** and **63**. Creating multiple keys allows multiple clients to use the BFD session.

- The secret data used to allow access to the session.
- The time at which the authentication key becomes active, in the format *yyyy-mm-dd.hh:mm:ss*.

```
[edit security authentication-key-chains key-chain bfd-rip]
user@R1# set key 53 secret $ABC123$ABC123
user@R1# set key 53 start-time "2012-2-16.12:00:00 -0800"
```

10. Configure tracing operations to track BFD authentication.

```
[edit protocols bfd traceoptions]
user@R1# set file bfd-trace
user@R1# set flag all
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
```

```
user@R1# show protocols
bfd {
    traceoptions {
        file bfd-trace;
        flag all;
    }
}
rip {
    group rip-group {
```

```
        export advertise-routes-through-rip;
        bfd-liveness-detection {
            minimum-interval 600;
        }
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

```
user@R1# show security
authentication-key-chains {
    key-chain bfd-rip {
        key 53 {
            secret $ABC123$ABC123
            start-time "2012-2-16.12:00:00 -0800";
        }
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying That the BFD Sessions Are Authenticated

Purpose

Make sure that the BFD sessions are authenticated.

Action

From operational mode, enter the `show bfd session detail` command.

```
user@R1> show bfd session detail
                                                Detect   Transmit
Address                  State     Interface    Time     Interval  Multiplier
10.0.0.2                 Up        fe-1/2/0.1   1.800     0.600        3
 Client RIP, TX interval 0.600, RX interval 0.600, Authenticate
 Session up time 01:39:34
 Local diagnostic None, remote diagnostic None
 Remote state Up, version 1
 Logical system 6, routing table index 53

1 sessions, 1 clients
Cumulative transmit rate 1.7 pps, cumulative receive rate 1.7 pps
```

Meaning

Authenticate is displayed to indicate that BFD authentication is configured.

Viewing Extensive Information About the BFD Authentication

Purpose

View the keychain name, the authentication algorithm and mode for each client in the session, and the BFD authentication configuration status.

Action

From operational mode, enter the `show bfd session extensive` command.

```
user@R1> show bfd session extensive
                                                Detect   Transmit
```

```
Address                  State     Interface      Time     Interval  Multiplier
10.0.0.2                 Up        fe-1/2/0.1     1.800     0.600        3
 Client RIP, TX interval 0.600, RX interval 0.600, Authenticate
        keychain bfd-rip, algo keyed-md5, mode loose
 Session up time 01:46:29
 Local diagnostic None, remote diagnostic None
 Remote state Up, version 1
 Logical system 6, routing table index 53
 Min async interval 0.600, min slow interval 1.000
 Adaptive async TX interval 0.600, RX interval 0.600
 Local min TX interval 0.600, minimum RX interval 0.600, multiplier 3
 Remote min TX interval 0.600, min RX interval 0.600, multiplier 3
 Local discriminator 225, remote discriminator 226
 Echo mode disabled/inactive
 Authentication enabled/active, keychain bfd-rip, algo keyed-md5, mode loose
  Session ID: 0x300501

1 sessions, 1 clients
Cumulative transmit rate 1.7 pps, cumulative receive rate 1.7 pps
```

Meaning

The output shows the keychain name, the authentication algorithm and mode for the client in the session, and the BFD authentication configuration status.

Checking the BFD Trace File

Purpose

Use tracing operations to verify that BFD packets are being exchanged.

Action

From operational mode, enter the `show log` command.

```
user@R1> show log bfd-trace
Feb 16 10:26:32 PPM Trace: BFD periodic xmit to 10.0.0.2 (IFL 124, rtbl 53, single-hop port)
Feb 16 10:26:32 Received Downstream TraceMsg (24) len 86:
Feb 16 10:26:32    IfIndex (3) len 4: 0
Feb 16 10:26:32    Protocol (1) len 1: BFD
Feb 16 10:26:32    Data (9) len 61: (hex) 42 46 44 20 70 61 63 6b 65 74 20 66 72 6f 6d 20 31 30
```

```
2e
Feb 16 10:26:32 PPM Trace: BFD packet from 10.0.0.1 (IFL 73, rtbl 56, ttl 255) absorbed
Feb 16 10:26:32 Received Downstream TraceMsg (24) len 60:
Feb 16 10:26:32    IfIndex (3) len 4: 0
Feb 16 10:26:32    Protocol (1) len 1: BFD
Feb 16 10:26:32    Data (9) len 35: (hex) 42 46 44 20 70 65 72 69 6f 64 69 63 20 78 6d 69 74 20
6f
...
```

Meaning

The output shows the normal functioning of BFD.

Change History Table

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

Release	Description
15.1X49	Note that the functionality of configuring BFD for RIP described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

Traffic Control in a RIP Network

IN THIS SECTION

Understanding Traffic Control with Metrics in a RIP Network

To tune a RIP network and control traffic flowing through the network, you increase or decrease the cost of the paths through the network. RIP provides two ways to modify the path cost: an incoming metric and an outgoing metric, which are each set to **1** by default. These metrics are attributes that manually specify the cost of any route advertised through a host. By increasing or decreasing the metrics—and thus the cost—of links throughout the network, you can control packet transmission across the network.

The incoming metric modifies the cost of an individual segment when a route across the segment is imported into the routing table. For example, if you set the incoming metric on the segment to **3**, the individual segment cost along the link is changed from **1** to **3**. The increased cost affects all route calculations through that link. Other routes that were previously excluded because of a high hop count might now be selected into the router's forwarding table.

The outgoing metric modifies the path cost for all the routes advertised out a particular interface. Unlike the incoming metric, the outgoing metric modifies the routes that other routers are learning and thereby controls the way they send traffic.

If an exported route was learned from a member of the same RIP group, the metric associated with that route is the normal RIP metric. For example, a RIP route with a metric of 5 learned from a neighbor configured with an incoming metric of 2 is advertised with a combined metric of 7 when advertised to neighbors in the same group. However, if this route was learned from a RIP neighbor in a different group or from a different protocol, the route is advertised with the metric value configured in the outgoing metric for that group.

You might want to increase the metric of routes to decrease the likelihood that a particular route is selected and installed in the routing table. This process is sometimes referred to as *route poisoning*. Some reasons that you might want to poison a route are that the route is relatively expensive to use, or it has relatively low bandwidth.

A route with a higher metric than another route becomes the active route only when the lower-metric route becomes unavailable. In this way, the higher-metric route serves as a backup path.

One way to increase the metric of imported routes is to configure an import policy. Another way is to include the `metric-in` statement in the RIP neighbor configuration. One way to increase the metric of export routes is to configure an export policy. Another way is to include the `metric-out` statement in the RIP neighbor configuration.

Example: Controlling Traffic in a RIP Network with an Incoming Metric

IN THIS SECTION

This example shows how to control traffic with an incoming metric.

Requirements

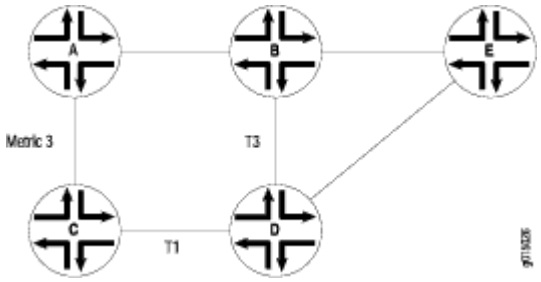
Before you begin, define RIP groups, and add interfaces to the groups. Then configure a routing policy to export directly connected routes and routes learned through the RIP routing exchanges. See "Example: Configuring a Basic RIP Network" on page 13.

Overview

IN THIS SECTION

In this example, routes to Router D are received by Router A across both of its RIP-enabled interfaces as shown in Figure 11 on page 86. Because the route through Router B and the route through Router C have the same number of hops, both routes are imported into the forwarding table. However, because the T3 link from Router B to Router D has a higher bandwidth than the T1 link from Router C to Router D, you want traffic to flow from Router A through Router B to Router D.

Figure 11: Controlling Traffic in a RIP Network with the Incoming Metric



Topology

To force this flow, you can modify the route metrics as they are imported into Router A's routing table. By setting the incoming metric on the interface from Router A to Router C, you modify the metric on all routes received through that interface. Setting the incoming route metric on Router A changes only the routes in Router A's routing table, and affects only how Router A sends traffic to Router D. Router D's route selection is based on its own routing table, which, by default, includes no adjusted metric values.

In the example, Router C receives a route advertisement from Router D and readvertises the route to Router A. When Router A receives the route, it applies the incoming metric on the interface. Instead of incrementing the metric by 1 (the default), Router A increments it by 3 (the configured incoming metric), giving the route from Router A to Router D through Router C a total path metric of 4. Because the route through Router B has a metric of 2, it becomes the preferred route for all traffic from Router A to Router D.

This example uses a RIP group called **alpha 1** on interface **g3–0/0/0**.

Configuration

IN THIS SECTION

Procedure

Step-by-Step Procedure

To control traffic with an incoming metric:

1. Enable RIP on the interface.

```
[edit protocols rip]
user@host# set group alpha1 neighbor ge-0/0/0
```

2. Set the incoming metric.

```
[edit protocols rip]
user@host# set metric-in 3
```

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

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

Verification

To verify that the configuration is working properly, enter the `show route protocols rip` command.

Example: Controlling Traffic in a RIP Network with an Outgoing Metric

IN THIS SECTION

This example shows how to control traffic with an outgoing metric.

Requirements

Before you begin:

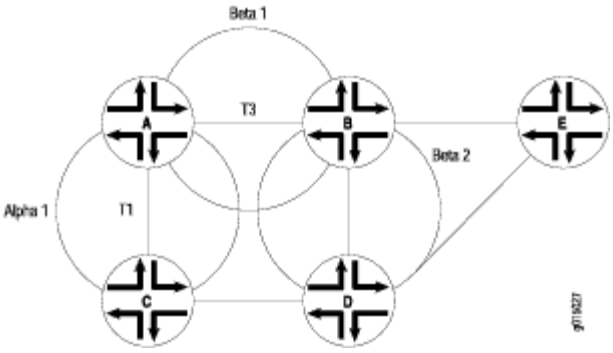
- Define RIP groups, and add interfaces to the groups. Then configure a routing policy to export directly connected routes and routes learned through RIP routing exchanges. See "Example: Configuring a Basic RIP Network" on page 13.
- Control traffic with an incoming metric. See "Example: Controlling Traffic in a RIP Network with an Incoming Metric" on page 85.

Overview

IN THIS SECTION

In this example, each route from Router A to Router D has two hops as shown in Figure 12 on page 88. However, because the link from Router A to Router B in the RIP group has a higher bandwidth than the link from Router A to Router C in RIP group Alpha 1, you want traffic from Router D to Router A to flow through Router B. To control the way Router D sends traffic to Router A, you can alter the routes that Router D receives by configuring the outgoing metric on Router A's interfaces in the Alpha 1 RIP group.

Figure 12: Controlling Traffic in a RIP Network with the Outgoing Metric



Topology

If the outgoing metric for the Alpha 1 RIP group—the A-to-C link—is changed to 3, Router D calculates the total path metric from Router A through Router C as 4. In contrast, the unchanged default total path metric to Router A through Router B in the RIP group is 2. The fact that Router A's interfaces belong to two different RIP groups allows you to configure two different outgoing metrics on its interfaces, because you configure path metrics at the group level.

By configuring the outgoing metric, you control the way Router A sends traffic to Router D. By configuring the outgoing metric on the same router, you control the way Router D sends traffic to Router A.

This example uses an outgoing metric of **3**.

Configuration

IN THIS SECTION

Procedure

Step-by-Step Procedure

To control traffic with an outgoing metric:

1. Set the outgoing metric.

```
[edit protocols rip group alpha1]
user@host# set metric-out 3
```

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

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

Verification

To verify that the configuration is working properly, enter the `show protocols rip` command.

Example: Configuring the Metric Value Added to Imported RIP Routes

IN THIS SECTION

This example shows how to change the default metric to be added to incoming routes to control the route selection process.

Requirements

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

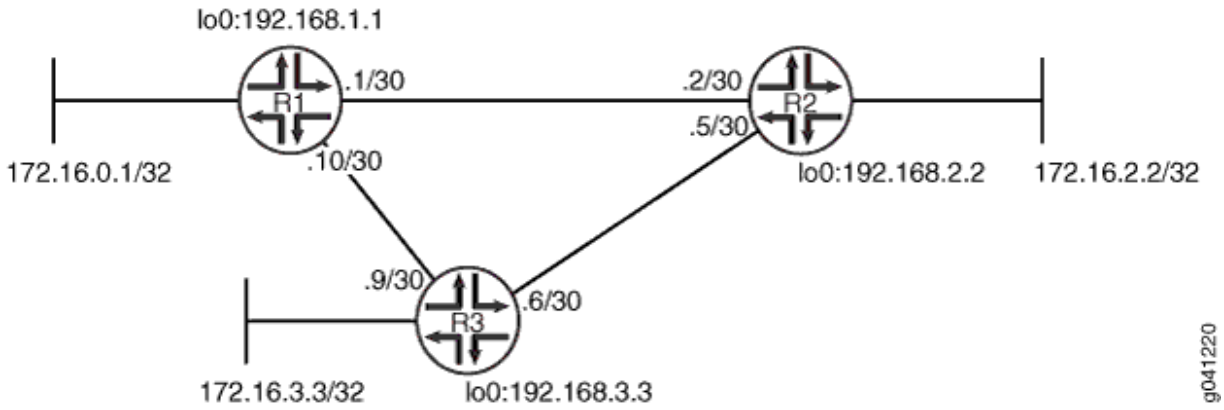
Overview

IN THIS SECTION

Normally, when multiple routes are available, RIP selects the route with the lowest hop count. Changing the default metric enables you to control the route selection process such that a route with a higher hop count can be preferred over of a route with a lower hop count.

Figure 13 on page 91 shows the topology used in this example.

Figure 13: RIP Incoming Metrics Network Topology



Device R1 has two potential paths to reach 172.16.2.2/32. The default behavior is to send traffic out the 0.1/30 interface facing Device R2. Suppose, though, that the path through Device R3 is less expensive to use or has higher bandwidth links. This example shows how to use the `metric-in` statement to ensure that Device R1 uses the path through Device R3 to reach 172.16.2.2/32. "CLI Quick Configuration" on page 91 shows the configuration for all of the devices in Figure 13 on page 91. The section "No Link Title" on page 93 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces ge-1/2/1 unit 10 description to-R3
set interfaces ge-1/2/1 unit 10 family inet address 10.0.0.10/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group primary export advertise-routes-through-rip
set protocols rip group primary neighbor ge-1/2/1.10
set protocols rip group secondary export advertise-routes-through-rip
set protocols rip group secondary neighbor fe-1/2/0.1 metric-in 4
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces ge-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor ge-1/2/1.5
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces ge-1/2/1 unit 9 family inet address 10.0.0.9/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set protocols rip group rip-group neighbor ge-1/2/1.9
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
```

```
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 a RIP metrics:

1. Configure the network interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 description to-R2
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
user@R1# set ge-1/2/1 unit 10 description to-R3
user@R1# set ge-1/2/1 unit 10 family inet address 10.0.0.10/30
user@R1# set lo0 unit 1 family inet address 172.16.0.1/32
user@R1# set lo0 unit 1 family inet address 192.168.1.1/32
```

2. Create the RIP groups and add the interfaces.

   To configure RIP in Junos OS, you must configure one or more groups that contain the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

   For the interface that is facing Device R2, the **metric-in 4** setting causes this route to be less likely to be chosen as the active route.

```
[edit protocols rip]
user@R1# set group primary neighbor ge-1/2/1.10
user@R1# set group secondary neighbor fe-1/2/0.1 metric-in 4
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip]
user@R1# set group primary export advertise-routes-through-rip
user@R1# set group secondary export advertise-routes-through-rip
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        description to-R2;
        family inet {
            address 10.0.0.1/30;
        }
    }
}
ge-1/2/1 {
    unit 10 {
        description to-R3;
        family inet {
            address 10.0.0.10/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.0.1/32;
            address 192.168.1.1/32;
        }
```

```
    }
}
```

```
user@R1# show protocols
rip {
    group primary {
        export advertise-routes-through-rip;
        neighbor ge-1/2/1.10;
    }
    group secondary {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.1 {
            metric-in 4;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying That the Expected Route Is Active

Purpose

Make sure that to reach 172.16.2.2/32, Device R1 uses the path through Device R3.

Action

From operational mode, enter the `show route 172.16.2.2` command.

```
user@R1> show route 172.16.2.2
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.2.2/32      *[RIP/100] 00:15:46, metric 3, tag 0
                    > to 10.0.0.9 via ge-1/2/1.10
```

Meaning

The **to 10.0.0.9 via ge-1/2/1.10** output shows that Device R1 uses the path through Device R3 to reach 172.16.2.2/32. The metric for this route is 3.

Removing the metric-in Statement

Purpose

Delete or deactivate the `metric-in` statement to see what happens to the 172.16.2.2/32 route.

Action

1. From configuration mode, deactivate the `metric-in` statement.

```
[edit protocols rip group secondary neighbor fe-1/2/0.1]
user@R1# deactivate metric-in
user@R1# commit
```

2. From operational mode, enter the `show route 172.16.2.2` command.

```
user@R1> show route 172.16.2.2
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.2.2/32      *[RIP/100] 00:00:06, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
```

Meaning

The **to 10.0.0.2 via fe-1/2/0.1** output shows that Device R1 uses the path through Device R2 to reach 172.16.2.2/32. The metric for this route is 2.

RELATED DOCUMENTATION

- RIP Configuration Overview | 12
- Verifying a RIP Configuration | 216
- RIP Overview | 2

Point-to-Multipoint RIP Networks

IN THIS SECTION

Configuring Point-to-Multipoint RIP Networks Overview

A point-to-multipoint RIP network consists of a device having two or more peers on a single interface. All the devices forming a point-to-multipoint connection are placed in a single broadcast domain.

In a RIP network, a device can have a single peer or multiple peers for an interface. However, the demand circuit feature implementation in a RIP network requires the use of a single RIP peer. When you configure the following statements, a RIP network with demand circuits can also be configured to have multiple peers on an interface:

- Configuring the interface type to be a multipoint interface by using the `interface-type (Protocols RIP) p2mp` statement.
- Enabling dynamic peer discovery by using the `dynamic-peers` statement (SRX Series Firewalls only).

> **NOTE**: Before configuring the `dynamic-peers` statement, IPsec must be configured and IPsec tunnels must be set up by configuring IPsec parameters. Without IPsec configuration, the remote peers have to be explicitly configured at the RIP protocol level by using the `peer` *address* statement. See Configuring Security Associations for IPsec on an ES PIC for more details.

- Configuring peers by using the `peer` *address* statement.

```
[edit]
protocols {
    rip {
        group red {
            neighbor fe-0/1/3 {
                interface-type (Protocols RIP) p2mp;
                peer address; (or use dynamic-peers;)
            }
        }
    }
}
```

The `show rip statistics peer` *address* command can be used to display the RIP statistics at the peer level. The `clear rip statistics peer` *address* command can be used to clear the RIP statistics for a peer. Alternatively, you can use the **show rip statistics peer all** and `clear rip statistics peer all` command to display and clear RIP statistics for all peers.

Example: Configuring Point-to-Multipoint RIP Networks

IN THIS SECTION

This example shows how to configure a point-to-multipoint RIP network.

Requirements

This example uses the following hardware and software components:

- M Series routers, MX Series routers, T Series routers, or SRX Series Firewalls
- Junos OS Release 12.1 or later

Overview

IN THIS SECTION

In a RIP network, a device can have a single peer or multiple peers for an interface. However, the demand circuit feature implementation in a RIP network requires the use of a single RIP peer.

When you include the following statements, the demand circuit implementation can have multiple peers for a given RIP neighbor.

- Configuring the interface type to be a multipoint interface by using the `interface-type (Protocols RIP) p2mp` statement.
- Enabling dynamic peer discovery by using the *dynamic-peers* statement (SRX Series Firewalls only).

NOTE: To configure the `dynamic-peers` statement, IPsec tunnels must be set up by configuring IPsec parameters. See Configuring Security Associations for IPsec on an ES PIC for more details.

- Configuring peers by using the `peer` *address* statement.

```
[edit]
protocols {
    rip {
        group red {
            neighbor fe-0/1/3 {
                interface-type (Protocols RIP) p2mp;
                peer address; (or use dynamic-peers;)
            }
        }
    }
}
```

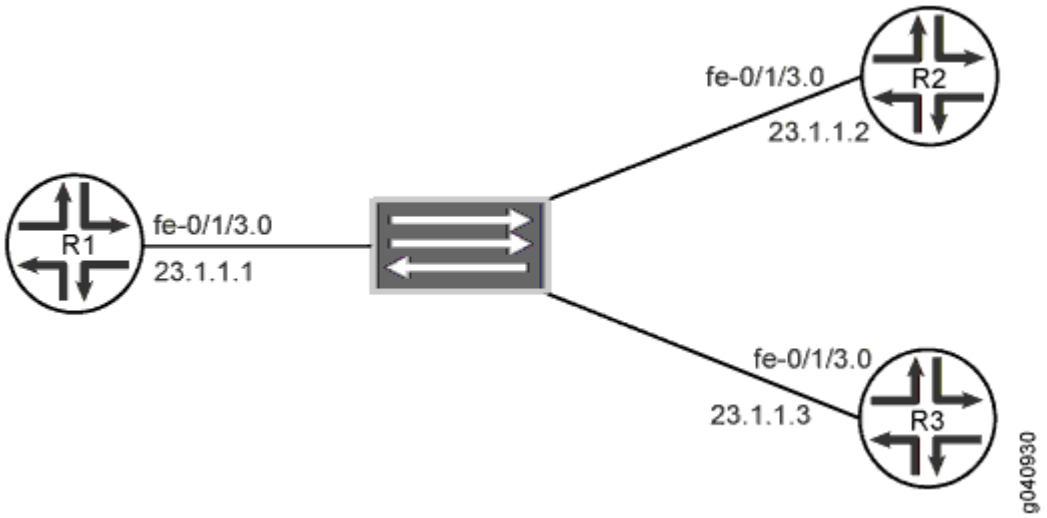
The `show rip statistics peer` command can be used to display the RIP statistics at the peer level.

Topology

In this example, Devices R1, R2, and R3 form a point-to-multipoint network. R1 is connected to R2 and to R3 as a point-to-multipoint connection through a switch that places all devices in the same broadcast domain. RIP demand circuits are configured on all three devices. The two peers to R1 are configured statically by using the `peer` *address* statement. The `dynamic-peers` statement is not used here.

Figure 14 on page 101 shows the topology used in this example.

Figure 14: Configuring a Point-to-Multipoint RIP Network



Configuration

IN THIS SECTION

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-0/1/3 unit 0 family inet address 23.1.1.1/24
set policy-options policy-statement accept-rip-routes term from-direct from protocol direct
set policy-options policy-statement accept-rip-routes term from-direct then accept
set policy-options policy-statement accept-rip-routes term from-rip from protocol rip
set policy-options policy-statement accept-rip-routes term from-rip then accept
set protocols rip traceoptions file R1.log size 4m world-readable
set protocols rip traceoptions flag all detail
set protocols rip group red export accept-rip-routes
```

```
set protocols rip group red neighbor fe-0/1/3.0 interface-type p2mp
set protocols rip group red neighbor fe-0/1/3.0 peer 23.1.1.2
set protocols rip group red neighbor fe-0/1/3.0 peer 23.1.1.3
set protocols rip group red neighbor fe-0/1/3.0 demand-circuit
set protocols rip group red neighbor fe-0/1/3.0 max-retrans-time 10
```

Similarly, configure Devices R2 and R3, omitting the **peer *address*** configuration statement.

Configuring a Point-to-Multipoint RIP Network (with Demand Circuits)

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 the point-to-multipoint feature across a RIP network:

1. Configure the device interface.

```
[edit interfaces fe-0/1/3 unit 0]
user@R1# set family inet address 23.1.1.1/24
```

2. Define a policy for exporting RIP routes from the routing table to the protocol for transmission through the network.

```
[edit policy-options policy-statement accept-rip-routes]
user@R1# set term from-direct from protocol direct
user@R1# set term from-direct then accept
user@R1# set term from-rip from protocol rip
user@R1# set term from-rip then accept
```

3. Configure RIP and a RIP group with the defined export policy and point-to-multipoint configuration statements.

```
[edit protocols rip]
user@R1# set traceoptions file R1.log size 4m world-readable
user@R1# set traceoptions flag all detail
user@R1# set group red export accept-rip-routes
user@R1# set group red neighbor fe-0/1/3.0 interface-type p2mp
```

```
user@R1# set group red neighbor fe-0/1/3.0 peer 23.1.1.2
user@R1# set group red neighbor fe-0/1/3.0 peer 23.1.1.3
user@R1# set group red neighbor fe-0/1/3.0 demand-circuit
user@R1# set group red neighbor fe-0/1/3.0 max-retrans-time 10
```

Similarly, configure Devices R2 and R3, omitting the **peer *address*** configuration statement.

NOTE: Configuring **max-retrans-time** is optional. In the absence of this configuration statement, the default retransmission time of 180 seconds is configured.

The configuration used in this example is for a RIP network with demand circuits. To configure RIP for networks without demand circuits, exclude the **demand-circuit** and `max-retrans-time` statements from the configuration and check the resulting output. For more information about configuring RIP demand circuits, see "Example: Configuring RIP Demand Circuits" on page 57.

Results

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

```
[edit]
user@R1# show interfaces
fe-0/1/3 {
    unit 0 {
        family inet {
            address 23.1.1.1/24;
        }
    }
}
```

```
user@R1# show protocols rip
traceoptions {
    file R1.log size 4m world-readable;
    flag all detail;
}
group red {
    export accept-rip-routes;
    neighbor fe-0/1/3.0 {
```

```
        interface-type p2mp;
        peer 23.1.1.2;
        peer 23.1.1.3;
        demand-circuit;
        max-retrans-time 10;
    }
}
```

```
user@R1# show policy-options
policy-statement accept-rip-routes {
    term from-direct {
        from protocol direct;
        then accept;
    }
    term from-rip {
        from protocol rip;
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

- Verifying the Point-to-Multipoint RIP Network | **104**

Confirm that the configuration is working properly.

Verifying the Point-to-Multipoint RIP Network

Purpose

Verify that the RIP network is functional with the point-to-multipoint feature configured.

Action

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

```
user@R1> show rip neighbor

                  Local  Source          Destination     Send   Receive   In
Neighbor          State  Address         Address         Mode   Mode     Met
--------          -----  -------         -----------     ----   -------  ---
fe-0/1/3.0(DC)       Up 23.1.1.1        23.1.1.2        unicast unicast    1
fe-0/1/3.0(DC)       Up 23.1.1.1        23.1.1.3        unicast unicast    1
```

From operational mode, run the `show rip statistics peer` *address* command.

```
user@R1> show rip statistics peer 23.1.1.2

RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              3              0              0              0

fe-0/1/3.0 Peer-Ip 23.1.1.2:  2 routes learned; 3 routes advertised; timeout 180s; update
interval 0s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                         0            0            0
Triggered Updates Sent               3            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received               2            0            0
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures              0            0            0
RIP Requests Received                0            0            0
```

```
RIP Requests Ignored                0           0           0
none                                3           0           0
```

```
user@R1> show rip statistics peer 23.1.1.3

RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              3              0              0              0

fe-0/1/3.0 Peer-Ip 23.1.1.3:  2 routes learned; 3 routes advertised; timeout 180s; update
interval 0s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                         0            0            0
Triggered Updates Sent               3            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
RIPv1 Updates Received               0            0            0
RIPv1 Bad Route Entries              0            0            0
RIPv1 Updates Ignored                0            0            0
RIPv2 Updates Received               2            0            0
RIPv2 Bad Route Entries              0            0            0
RIPv2 Updates Ignored                0            0            0
Authentication Failures              0            0            0
RIP Requests Received                0            0            0
RIP Requests Ignored                 0            0            0
none                                 3            0            0
```

Meaning

The RIP network is up and running with the point-to-multipoint feature configured.

RELATED DOCUMENTATION

dynamic-peers

Example: Configuring RIP Demand Circuits | **57**

interface-type (Protocols RIP)

peer

RIP Import Policy

IN THIS SECTION

Understanding RIP Import Policy

The default RIP import policy is to accept all received RIP routes that pass a sanity check. To filter routes being imported by the local routing device from its neighbors, include the `import` statement, and list the names of one or more policies to be evaluated. If you specify more than one policy, they are evaluated in order (first to last) and the first matching policy is applied to the route. If no match is found, the local routing device does not import any routes. Note that the functionality of applying policies to RIP routes imported from neighbors described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

Example: Applying Policies to RIP Routes Imported from Neighbors

IN THIS SECTION

This example shows how to configure an import policy in a RIP network.

Requirements

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

Overview

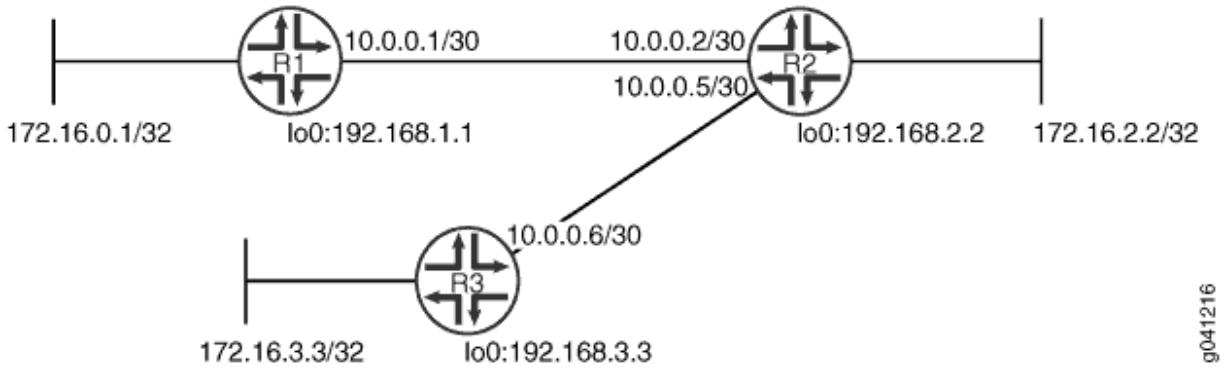
IN THIS SECTION

In this example, Device R1 has an import policy that accepts the 10/8 and 192.168/16 RIP routes and rejects all other RIP routes. This means that the 172.16/16 RIP routes are excluded from Device R1's routing table.

An export policy is also shown because an export policy is required as part of the minimum configuration for RIP.

Figure 15 on page 108 shows the topology used in this example.

Figure 15: RIP Import Policy Network Topology



"CLI Quick Configuration" on page 109 shows the configuration for all of the devices in Figure 15 on page 108. The section "No Link Title" on page 110 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip import rip-import
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
set policy-options policy-statement rip-import term 1 from protocol rip
set policy-options policy-statement rip-import term 1 from route-filter 10.0.0.0/8 orlonger
set policy-options policy-statement rip-import term 1 from route-filter 192.168.0.0/16 orlonger
set policy-options policy-statement rip-import term 1 then accept
set policy-options policy-statement rip-import term 2 then reject
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 a RIP import policy:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
   user@R1# set lo0 unit 1 family inet address 172.16.0.1/32
   user@R1# set lo0 unit 1 family inet address 192.168.1.1/32
   ```

2. Create the RIP group and add the interface.

   To configure RIP in Junos OS, you must configure a group that contains the interfaces on which RIP is enabled.

   You do not need to enable RIP on the loopback interface.

   ```
   [edit protocols rip group rip-group]
   user@R1# set neighbor fe-1/2/0.1
   ```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

5. Configure the import policy.

```
[edit policy-options policy-statement rip-import]
user@R1# set term 1 from protocol rip
user@R1# set term 1 from route-filter 10.0.0.0/8 orlonger
user@R1# set term 1 from route-filter 192.168.0.0/16 orlonger
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

6. Apply the import policy.

```
[edit protocols rip]
user@R1# set import rip-import
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
```

```
            address 10.0.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.0.1/32;
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
rip {
    import rip-import;
    group rip-group {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
policy-statement rip-import {
    term 1 {
        from {
            protocol rip;
            route-filter 10.0.0.0/8 orlonger;
            route-filter 192.168.0.0/16 orlonger;
        }
        then accept;
    }
    term 2 {
        then reject;
```

```
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Looking at the Routes That Device R2 Is Advertising to Device R1

Purpose

Verify that Device R2 is sending the expected routes.

Action

From operational mode, enter the `show route advertising-protocol rip` command.

```
user@R2> show route advertising-protocol rip 10.0.0.2

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[Direct/0] 2d 01:17:44
                    >    via fe-1/2/0.5
172.16.2.2/32      *[Direct/0] 2d 04:09:52
                    >    via lo0.2
172.16.3.3/32      *[RIP/100] 23:40:02, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
192.168.2.2/32     *[Direct/0] 2d 04:09:52
```

```
                    >    via lo0.2
192.168.3.3/32     *[RIP/100] 23:40:02, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
```

Meaning

Device R2 is sending 172.16/16 routes to Device R1.

Looking at the Routes That Device R1 Is Receiving from Device R2

Purpose

Verify that Device R1 is receiving the expected routes.

Action

From operational mode, enter the `show route receive-protocol rip` command.

```
user@R1> show route receive-protocol rip 10.0.0.2
inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[RIP/100] 01:06:03, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.2.2/32     *[RIP/100] 01:06:03, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.3.3/32     *[RIP/100] 01:06:03, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
```

Meaning

The output shows that the 172.16/16 routes are excluded.

Checking the Routing Table

Purpose

Verify that the routing table is populated with the expected routes.

Action

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

```
user@R1> show route protocol rip

inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.4/30        *[RIP/100] 00:54:34, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.2.2/32     *[RIP/100] 00:54:34, metric 2, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
192.168.3.3/32     *[RIP/100] 00:54:34, metric 3, tag 0
                    > to 10.0.0.2 via fe-1/2/0.1
224.0.0.9/32       *[RIP/100] 00:49:00, metric 1
                          MultiRecv
```

Meaning

The output shows that the routes have been learned from Device R2 and Device R3.

If you delete or deactivate the import policy, the routing table contains the 172.16/16 routes.

Testing the Import Policy

Purpose

By using the `test policy` command, monitor the number of rejected prefixes.

Action

From operational mode, enter the `test policy rip-import 172.16/16` command.

```
user@R1> test policy rip-import 172.16/16
Policy rip-import: 0 prefix accepted, 1 prefix rejected
```

Meaning

The output shows that the policy rejected one prefix.

Change History Table

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

Release	Description
15.1X49	Note that the functionality of applying policies to RIP routes imported from neighbors described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

Interoperability of RIPv1 and RIPv2 Networks

IN THIS SECTION

Understanding the Sending and Receiving of RIPv1 and RIPv2 Packets

RIP version 1 (RIPv1) and RIP version 2 (RIPv2) can run simultaneously. This might make sense when you are migrating a RIPv1 network to a RIPv2 network. This also allows interoperation with a device that supports RIPv1 but not RIPv2.

By default, when RIP is enabled on an interface, Junos OS receives both RIPv1 and RIPv2 packets and sends only RIPv2 packets. You can configure this behavior by including the **send** and `receive` statements in the RIP configuration.Note that the functionality of configuring the sending and receiving of RIPv1 and RIPv2 packets described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

Example: Configuring the Sending and Receiving of RIPv1 and RIPv2 Packets

IN THIS SECTION

This example shows how to configure whether the RIP update messages conform to RIP version 1 (RIPv1) only, to RIP version 2 (RIPv2) only, or to both versions. You can also disable the sending or receiving of update messages.

Requirements

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

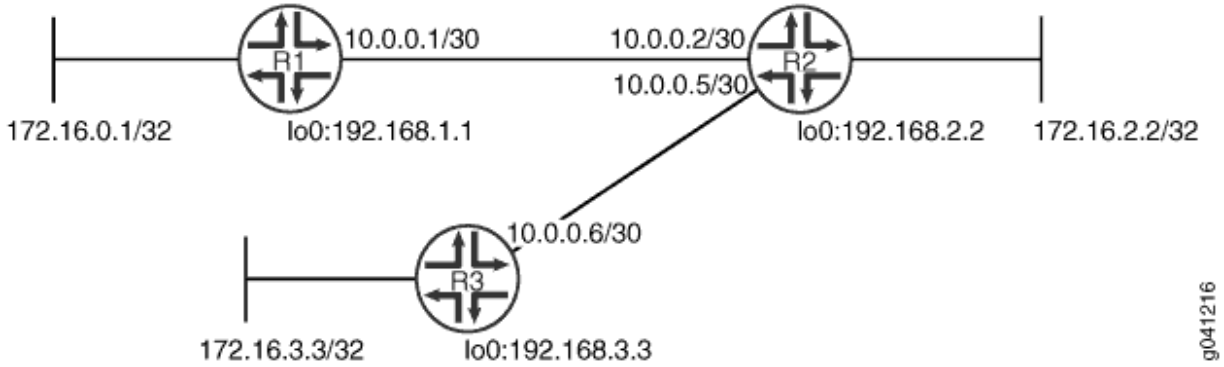
Overview

IN THIS SECTION

By default, when RIP is enabled on an interface, Junos OS receives both RIPv1 and RIPv2 packets and sends only RIPv2 packets.

Figure 16 on page 118 shows the topology used in this example.

Figure 16: Sending and Receiving RIPv1 and RIPv2 Packets Network Topology



In this example, Device R1 is configured to receive only RIPv2 packets.

"CLI Quick Configuration" on page 118 shows the configuration for all of the devices in Figure 16 on page 118. The section "No Link Title" on page 119 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

- Procedure | **118**

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group rip-group export advertise-routes-through-rip
```

```
set protocols rip group rip-group neighbor fe-1/2/0.1 receive version-2
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.2
set protocols rip group rip-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group rip-group export advertise-routes-through-rip
set protocols rip group rip-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 a RIP packet versions that can be received:

1. Configure the network interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
```

```
user@R1# set lo0 unit 1 family inet address 172.16.0.1/32
user@R1# set lo0 unit 1 family inet address 192.168.1.1/32
```

2. Create the RIP groups and add the interfaces.

   To configure RIP in Junos OS, you must configure one or more groups that contain the interfaces on which RIP is enabled. You do not need to enable RIP on the loopback interface.

   For the interface that is facing Device R2, the **receive version-2** setting causes this interface to accept only RIPv2 packets.

```
[edit protocols rip group rip-group]
user@R1# set neighbor fe-1/2/0.1 receive version-2
```

3. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R1# set from protocol direct
user@R1# set from protocol rip
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group rip-group]
user@R1# set export advertise-routes-through-rip
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
```

```
        }
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.0.1/32;
            address 192.168.1.1/32;
        }
    }
}
```

```
user@R1# show protocols
rip {
    group rip-group {
        export advertise-routes-through-rip;
        neighbor fe-1/2/0.1 {
            receive version-2;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

- Verifying That the Receive Mode Is Set to RIPv2 Only | 122

Confirm that the configuration is working properly.

Verifying That the Receive Mode Is Set to RIPv2 Only

Purpose

Make sure that the interfacing Device R2 is configured to receive only RIPv2 packets, instead of both RIPv1 and RIPv2 packets.Starting in Junos OS Release 19.3R1, Junos OS supports RIP version 2 (RIPv2) for both IPv4 and IPv6 packets on ACX5448 Universal Metro routers.

Action

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

```
user@R1> show rip neighbor

                 Local  Source          Destination     Send   Receive   In
Neighbor         State  Address         Address         Mode   Mode     Met
--------         -----  -------         -----------     ----   -------  ---
fe-1/2/0.1           Up 10.0.0.1        224.0.0.9       mcast  v2 only    1
```

Meaning

In the output, the **Receive Mode** field displays **v2 only**. The default **Receive Mode** is **both**.

Change History Table

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

Release	Description
19.3R1	Starting in Junos OS Release 19.3R1, Junos OS supports RIP version 2 (RIPv2) for both IPv4 and IPv6 packets on ACX5448 Universal Metro routers.
15.149	Note that the functionality of configuring the sending and receiving of RIPv1 and RIPv2 packets described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

Route Redistribution Between RIP Instances

IN THIS SECTION

Understanding Route Redistribution Among RIP Instances

You can redistribute routes among RIP processes. Another way to say this is to export RIP routes from one RIP instance to other RIP instances.

In Junos OS, route redistribution among routing instances is accomplished by using routing table groups, also called RIB groups. Routing table groups allow you to import and export routes from a protocol within one routing table into another routing table. Note that the functionality of redistributing routes among RIP instances described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

NOTE: In contrast, the policy-based import and export functions allow you import and export routes between different protocols within the same routing table.

Consider the following partial example:

```
protocols {
    rip {
        rib-group inet-to-voice;
    }
}
routing-instances {
    voice {
        protocols {
            rip {
                rib-group voice-to-inet;
            }
        }
    }
```

```
}
routing-options {
    rib-groups {
        inet-to-voice {
            import-rib [ inet.0 voice.inet.0 ];
        }
        voice-to-inet {
            import-rib [ voice.inet.0 inet.0 ];
        }
    }
}
```

The way to read the `import-rib` statement is as follows. Take the routes from the protocol (RIP, in this case), and import them into the primary (or local) routing table and also into any other routing tables listed after this. The primary routing table is the routing table where the routing table group is being used. That would be either **inet.0** if used in the main routing instance or **voice.inet.0** if used within the routing instance. In the **inet-to-voice** routing table group, **inet.0** is listed first because this routing table group is used in the main routing instance. In the **voice-to-inet** routing table group, **voice.inet.0** is listed first because this routing table group is used in the voice routing instance.

Example: Redistributing Routes Between Two RIP Instances

IN THIS SECTION

This example shows how to configure a RIP routing instance and control the redistribution of RIP routes between the routing instance and the primary instance.

Requirements

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

Overview

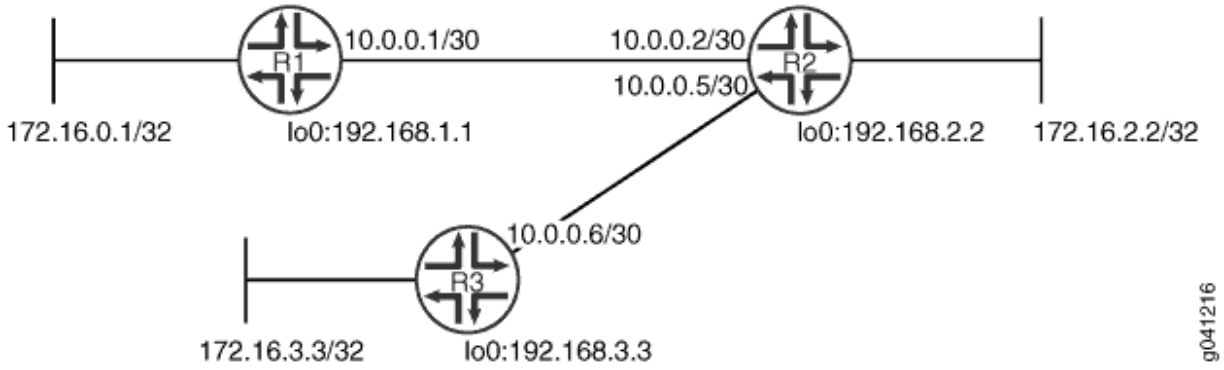
IN THIS SECTION

- Topology | 125

When you create a routing instance called voice, Junos OS creates a routing table called **voice.inet.0**. The example shows how to install routes learned through the primary RIP instance into the **voice.inet.0** routing table. The example also shows how to install routes learned through the voice routing instance into **inet.0**. This is done by configuring routing table groups. RIP routes are installed into each routing table that belongs to a routing table group.

Figure 17 on page 125 shows the topology used in this example.

Figure 17: Redistributing Routes Between RIP Instances Network Topology



"CLI Quick Configuration" on page 126 shows the configuration for all of the devices in Figure 17 on page 125. The section "No Link Title" on page 127 describes the steps on Device R2.

Topology

Configuration

IN THIS SECTION

- Procedure | 126

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces lo0 unit 1 family inet address 172.16.0.1/32
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols rip group to-R2 export advertise-routes-through-rip
set protocols rip group to-R2 neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 5 family inet address 10.0.0.5/30
set interfaces lo0 unit 2 family inet address 192.168.2.2/32
set interfaces lo0 unit 2 family inet address 172.16.2.2/32
set protocols rip rib-group inet-to-voice
set protocols rip group to-R3 export advertise-routes-through-rip
set protocols rip group to-R3 neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
set routing-instances voice protocols rip group to-R1 export advertise-routes-through-rip
set routing-instances voice interface fe-1/2/0.2
set routing-instances voice protocols rip rib-group voice-to-inet
set routing-instances voice protocols rip group to-R1 neighbor fe-1/2/0.2
set routing-options rib-groups inet-to-voice import-rib inet.0
set routing-options rib-groups inet-to-voice import-rib voice.inet.0
set routing-options rib-groups voice-to-inet import-rib voice.inet.0
set routing-options rib-groups voice-to-inet import-rib inet.0
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 3 family inet address 192.168.3.3/32
set interfaces lo0 unit 3 family inet address 172.16.3.3/32
set protocols rip group to-R2 export advertise-routes-through-rip
set protocols rip group to-R2 neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-rip term 1 from protocol rip
set policy-options policy-statement advertise-routes-through-rip term 1 then accept
```

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 redistribute RIP routes between routing instances:

1. Configure the network interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 2 family inet address 10.0.0.2/30
user@R2# set fe-1/2/1 unit 5 family inet address 10.0.0.5/30
user@R2# set lo0 unit 2 family inet address 192.168.2.2/32
user@R2# set lo0 unit 2 family inet address 172.16.2.2/32
```

2. Create the routing instance, and add one or more interfaces to the routing instance.

```
[edit routing-instances voice]
user@R2# set interface fe-1/2/0.2
```

3. Create the RIP groups and add the interfaces.

```
[edit protocols rip group to-R3]
user@R2# set neighbor fe-1/2/1.5
[edit routing-instances voice protocols rip group to-R1]
user@R2# set neighbor fe-1/2/0.2
```

4. Create the routing table groups.

```
[edit routing-options rib-groups]
user@R2# set inet-to-voice import-rib inet.0
user@R2# set inet-to-voice import-rib voice.inet.0
user@R2# set voice-to-inet import-rib voice.inet.0
user@R2# set voice-to-inet import-rib inet.0
```

5. Apply the routing table groups.

```
[edit protocols rip]
user@R2# set rib-group inet-to-voice
[edit routing-instances voice protocols rip]
user@R2# set rib-group voice-to-inet
```

6. Create the routing policy to advertise both direct and RIP-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-rip term 1]
user@R2# set from protocol direct
user@R2# set from protocol rip
user@R2# set then accept
```

7. Apply the routing policy.

   In Junos OS, you can only apply RIP export policies at the group level.

```
[edit protocols rip group to-R3]
user@R2# set export advertise-routes-through-rip
[edit routing-instances voice protocols rip group to-R1]
user@R2# set export advertise-routes-through-rip
```

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 configuration instructions in this example to correct it.

```
user@R2# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 5 {
        family inet {
            address 10.0.0.5/30;
        }
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.168.2.2/32;
            address 172.16.2.2/32;
        }
    }
}
```

```
user@R2# show protocols
rip {
    rib-group inet-to-voice;
    group to-R3 {
        export advertise-routes-through-rip;
        neighbor fe-1/2/1.5;
```

```
    }
}
```

```
user@R2# show policy-options
policy-statement advertise-routes-through-rip {
    term 1 {
        from protocol [ direct rip ];
        then accept;
    }
}
```

```
user@R2# show routing-instances
voice {
    interface fe-1/2/0.2;
    protocols {
        rip {
            rib-group voice-to-inet;
            group to-R1 {
                export advertise-routes-through-rip;
                neighbor fe-1/2/0.2;
            }
        }
    }
}
```

```
user@R2# show routing-options
rib-groups {
    inet-to-voice {
        import-rib [ inet.0 voice.inet.0 ];
    }
    voice-to-inet {
        import-rib [ voice.inet.0 inet.0 ];
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the Routing Tables

Purpose

Make sure that the routing tables contain the expected routes.

Action

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

```
user@R2> show route protocol rip
inet.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.1/32      *[RIP/100] 01:58:14, metric 2, tag 0
                    > to 10.0.0.1 via fe-1/2/0.2
172.16.3.3/32      *[RIP/100] 02:06:03, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
192.168.1.1/32     *[RIP/100] 01:58:14, metric 2, tag 0
                    > to 10.0.0.1 via fe-1/2/0.2
192.168.3.3/32     *[RIP/100] 02:06:03, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
224.0.0.9/32       *[RIP/100] 01:44:13, metric 1
                          MultiRecv

voice.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.1/32      *[RIP/100] 02:06:03, metric 2, tag 0
                    > to 10.0.0.1 via fe-1/2/0.2
```

```
172.16.3.3/32      *[RIP/100] 01:58:14, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
192.168.1.1/32     *[RIP/100] 02:06:03, metric 2, tag 0
                    > to 10.0.0.1 via fe-1/2/0.2
192.168.3.3/32     *[RIP/100] 01:58:14, metric 2, tag 0
                    > to 10.0.0.6 via fe-1/2/0.5
224.0.0.9/32       *[RIP/100] 01:44:13, metric 1
                          MultiRecv
```

Meaning

The output shows that both routing tables contain all of the RIP routes.

Change History Table

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

Release	Description
15.1X49	Note that the functionality of redistributing routes among RIP instances described in this topic is not supported in Junos OS Releases 15.1X49, 15.1X49-D30, or 15.1X49-D40.

3

CHAPTER

Configuring RIPng

IN THIS CHAPTER

Basic RIPng Configuration

IN THIS SECTION

Understanding Basic RIPng Routing

By default, RIP next generation (RIPng) routes are not redistributed. You must configure export policy to redistribute RIPng routes.

To have a router exchange routes with other routers, you must configure RIPng groups and neighbors. RIPng routes received from routers not configured as RIPng neighbors are ignored. Likewise, RIPng routes are advertised only to routers configured as RIPng neighbors.

Example: Configuring a Basic RIPng Network

IN THIS SECTION

This example shows how to configure a basic RIPng network.

Requirements

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

Overview

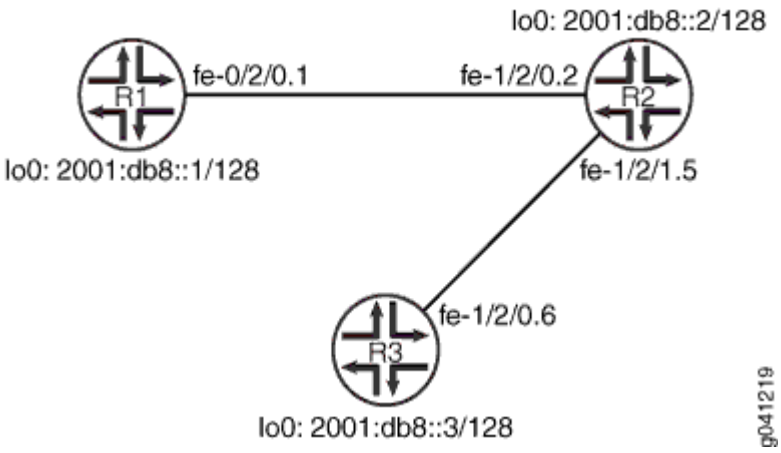
IN THIS SECTION

In this example, you configure a basic RIPng network, create a RIPng group called **ripng-group**, and add the directly connected interfaces to the RIPng group. Then you configure a routing policy to advertise direct routes using the policy statement **advertise-routes-through-ripng**.

By default, Junos OS does not advertise RIPng routes, not even routes that are learned through RIPng. To advertise RIPng routes, you must configure and apply an export routing policy that advertises RIPng-learned and direct routes.

To use RIPng on the device, you must configure RIPng on all of the RIPng interfaces within the network. Figure 18 on page 135 shows the topology used in this example.

Figure 18: Sample RIPng Network Topology



"CLI Quick Configuration" on page 136 shows the configuration for all of the devices in Figure 18 on page 135. The section "No Link Title" on page 137 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 description to-R1
set interfaces fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
set interfaces fe-1/2/1 unit 5 description to-R3
set interfaces fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
set interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.2
set protocols ripng group ripng-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 description to-R2
set interfaces fe-1/2/0 unit 6 family inet6 address 2001:db8:0:4::/64 eui-64
set interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

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 a basic RIPng network:

1. Configure the network interfaces.

   Use the `eui-64` statement to automatically generate the host portion of the interface address and the link-local address.

   For the loopback interface, you must assign a 128-bit address.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 unit 1 description to-R2
   user@R1# set fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
   user@R1# set lo0 unit 1 family inet6 address 2001:db8::1/128
   ```

2. Create the RIPng group and add the interface.

   To configure RIPng in Junos OS, you must configure a group that contains the interfaces on which RIPng is enabled. You do not need to enable RIPng on the loopback interface.

   ```
   [edit protocols ripng group ripng-group]
   user@R1# set neighbor fe-1/2/0.1
   ```

3. Create the routing policy to advertise both direct and RIPng-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-ripng term 1]
user@R1# set from protocol direct
user@R1# set from protocol ripng
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIPng export policies at the group level.

```
[edit protocols ripng group ripng-group]
user@R1# set export advertise-routes-through-ripng
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        description to-R2;
        family inet6 {
            address 2001:db8:0:1::/64 {
                eui-64;
            }
        }
    }
}
lo0 {
    unit 1 {
        family inet6 {
            address 2001:db8::1/128;
        }
```

```
    }
}
```

```
user@R1# show protocols
ripng {
    group ripng-group {
        export advertise-routes-through-ripng;
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-ripng {
    term 1 {
        from protocol [ direct ripng ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the Routing Table

Purpose

Verify that the routing table is populated with the expected routes.

Action

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

```
user@R1> show route protocol ripng
inet6.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::2/128    *[RIPng/100] 3d 19:24:43, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8::3/128    *[RIPng/100] 3d 19:24:40, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:2::/64  *[RIPng/100] 3d 19:24:43, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:3::/64  *[RIPng/100] 3d 19:24:43, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:4::/64  *[RIPng/100] 3d 19:24:40, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
ff02::9/128        *[RIPng/100] 3d 19:24:47, metric 1
                          MultiRecv
```

Meaning

The output shows that the routes have been learned from Device R2 and Device R3.

If you were to delete the **from protocol ripng** condition in the routing policy on Device R2, the remote routes from Device R3 would not be learned on Device R1.

Checking the Interface Addresses

Purpose

Verify that the `eui-64` statement automatically generated the host portion of the interface address and the link-local address.

Action

From operational mode, enter the `show interfaces terse` command.

```
user@R1> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
fe-1/2/0
fe-1/2/0.1              up    up   inet6    2001:db8:0:1:2a0:a514:0:14c/64
                                            fe80::2a0:a514:0:14c/64
lo0
lo0.1                   up    up   inet6    2001:db8::1
                                            fe80::2a0:a50f:fc56:14c
```

Meaning

The output shows that the interface address on fe-1/2/0.1 includes both the network portion (2001:db8:0:1) and the host portion (2a0:a514:0:14c).

Also, link-local (fe80) addresses are assigned to interfaces fe-1/2/0.1 and lo0.1.

Looking at the Routes That Device R1 Is Advertising to Device R2

Purpose

Verify that Device R1 is sending the expected routes.

Action

From operational mode, enter the `show route advertising-protocol ripng` command, using Device R1's link-local address as the neighbor address.

```
user@R1> show route advertising-protocol ripng fe80::2a0:a514:0:14c
inet6.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::1/128    *[Direct/0] 3d 19:45:55
                    >    via lo0.1
2001:db8:0:1::/64  *[Direct/0] 3d 19:45:55
                    >    via fe-1/2/0.1
```

Meaning

Device R1 is sending routes to its directly connected networks.

Verifying the RIPng-Enabled Interfaces

Purpose

Verify that all RIPng-enabled Interfaces are available and active.

Action

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

```
user@R1> show ripng neighbor
                    Source                          Dest              In
Neighbor     State  Address                         Address  Send Recv Met
--------     -----  -------                         -------  ---- ---- ---
fe-1/2/0.1      Up fe80::2a0:a514:0:14c             ff02::9  yes  yes  1
```

Meaning

The output shows that the RIPng-enabled interface on Device R1 is operational.

The output also shows the link-local address that is assigned to Device R2's directly connected link-local interface.

In general for this command, the output shows a list of the RIPng neighbors that are configured on the device. Verify the following information:

- Each configured interface is present. Interfaces are listed in alphabetical order.
- Each configured interface is up. The state of the interface is listed in the **State** column. A state of **Up** indicates that the link is passing RIPng traffic. A state of **Dn** indicates that the link is not passing RIPng traffic. In a point-to-point link, this state generally means that either the end point is not configured for RIPng or the link is unavailable.

Looking at the Routes That Device R1 Is Receiving from Device R2

Purpose

Verify that Device R1 is receiving the expected routes.

Action

From operational mode, enter the `show route receive-protocol ripng` command, using Device R2's directly connected link-local interface address as the neighbor address.

```
user@R1> show route receive-protocol ripng fe80::2a0:a514:0:24c
inet6.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::2/128    *[RIPng/100] 3d 19:58:09, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8::3/128    *[RIPng/100] 3d 19:58:06, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:2::/64  *[RIPng/100] 3d 19:58:09, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:3::/64  *[RIPng/100] 3d 19:58:09, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8:0:4::/64  *[RIPng/100] 3d 19:58:06, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
```

Meaning

Device R1 is receiving from Device R2 all of Device R2's directly connected networks. Device R1 is also receiving from Device R2 all of Device R3's directly connected networks, which Device R2 learned from Device R3 through RIPng.

Verifying the Exchange of RIPng Messages

Purpose

Verify that RIPng messages are being sent and received on all RIPng-enabled interfaces.

Action

From operational mode, enter the `show ripng statistics` command.

```
user@R1> show ripng statistics
RIPng info: port 521; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0
```

```
fe-1/2/0.1:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                    11632           10            2
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
Updates Received                11634           11            2
Bad Route Entries                   0            0            0
Updates Ignored                     0            0            0
RIPng Requests Received             1            0            0
RIPng Requests Ignored              0            0            0
```

Meaning

The output shows the number of RIPng routes learned. It also shows the number of RIPng updates sent and received on the RIPng-enabled interfaces. Verify the following information:

- The number of RIPng routes learned matches the number of expected routes learned. Subnets learned by direct connectivity through an outgoing interface are not listed as RIPng routes.
- RIPng updates are being sent on each RIPng-enabled interface. If no updates are being sent, the routing policy might not be configured to export routes.
- RIPng updates are being received on each RIPng-enabled interface. If no updates are being received, the routing policy might not be configured to export routes on the host connected to that subnet. The lack of updates might also indicate an authentication error.

Verifying Reachability of All Hosts in the RIPng Network

Purpose

By using the traceroute command on each loopback address in the network, verify that all hosts in the RIPng network are reachable from each Juniper Networks device.

Action

From operational mode, enter the `traceroute` command.

```
user@R1> traceroute 2001:db8::3
traceroute6 to 2001:db8::3 (2001:db8::3) from 2001:db8:0:1:2a0:a514:0:14c, 64 hops max, 12 byte
```

```
packets
 1  2001:db8:0:2:2a0:a514:0:24c (2001:db8:0:2:2a0:a514:0:24c)  8.881 ms  1.175 ms  1.101 ms
 2  2001:db8::3 (2001:db8::3)  1.544 ms  2.445 ms  2.043 ms
```

Meaning

Each numbered row in the output indicates a routing hop in the path to the host. The three-time increments indicate the round-trip time (RTT) between the device and the hop for each traceroute packet.

To ensure that the RIPng network is healthy, verify the following information:

- The final hop in the list is the host you want to reach.
- The number of expected hops to the host matches the number of hops in the traceroute output. The appearance of more hops than expected in the output indicates that a network segment is probably unreachable. It might also indicate that the incoming or outgoing metric on one or more hosts has been set unexpectedly.

RELATED DOCUMENTATION

RIPng Overview | 8

RIPng Import Policy

IN THIS SECTION

Understanding RIPng Import Policies to Filter Routes

The default RIPng import policy is to accept all received RIPng routes that pass a validity check. To filter routes being imported by the local routing device from its neighbors, include the `import` statement and list the names of one or more policies to be evaluated. If you specify more than one policy, they are evaluated in order (first to last) and the first matching policy is applied to the route. If no match is found, the local routing device does not import any routes.

Example: Applying Policies to RIPng Routes Imported from Neighbors

IN THIS SECTION

This example shows how to configure an import policy in a RIPng network.

Requirements

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

Overview

IN THIS SECTION

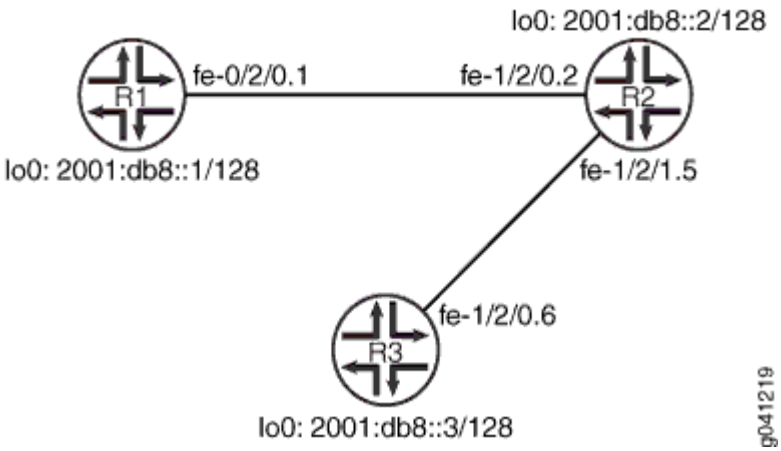
In this example, Device R2 has several extra loopback interface addresses configured to simulate additional networks.

Device R1 has an import policy that accepts the fe80::/64 and 2001:db8::/64 routes and rejects all other routes. This means that the extra networks advertised by Device R2 are not accepted into Device R1's routing table.

An export policy is also shown because an export policy is required as part of the minimum configuration for RIPng.

Figure 19 on page 147 shows the topology used in this example.

Figure 19: RIPng Import Policy Network Topology



"CLI Quick Configuration" on page 148 shows the configuration for all of the devices in Figure 19 on page 147. The section "No Link Title" on page 149 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

- Procedure | **148**

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.1 import ripng-import
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
set policy-options policy-statement ripng-import term 1 from route-filter fe80::/64 orlonger
set policy-options policy-statement ripng-import term 1 from route-filter 2001:db8::/64 orlonger
set policy-options policy-statement ripng-import term 1 then accept
set policy-options policy-statement ripng-import term 2 then reject
```

Device R2

```
set interfaces fe-1/2/0 unit 2 description to-R1
set interfaces fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
set interfaces fe-1/2/1 unit 5 description to-R3
set interfaces fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
set interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
set interfaces lo0 unit 2 family inet6 address 2002:db8::2/128
set interfaces lo0 unit 2 family inet6 address 2002:db9::2/128
set interfaces lo0 unit 2 family inet6 address 2002:db7::2/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.2
set protocols ripng group ripng-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R3

```
set  interfaces fe-1/2/0 unit 6 description to-R2
set  interfaces fe-1/2/0 unit 6 family inet6 address 2001:db8:0:4::/64 eui-64
set  interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

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 a RIPng import policy:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 unit 1 description to-R2
   user@R1# set fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
   user@R1# set lo0 unit 1 family inet6 address 2001:db8::1/128
   ```

2. Create the RIPng group and add the interface.

   To configure RIPng in Junos OS, you must configure a group that contains the interfaces on which RIPng is enabled. You do not need to enable RIPng on the loopback interface.

   ```
   [edit protocols ripng group ripng-group]
   user@R1# set neighbor fe-1/2/0.1
   ```

3. Create the routing policy to advertise both direct and RIPng-learned routes.

   ```
   [edit policy-options policy-statement advertise-routes-through-ripng term 1]
   user@R1# set from protocol direct
   ```

```
user@R1# set from protocol ripng
user@R1# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIPng export policies at the group level.

```
[edit protocols ripng group ripng-group]
user@R1# set export advertise-routes-through-ripng
```

5. Configure the import policy.

```
[edit policy-options policy-statement ripng-import]
user@R1# set term 1 from route-filter fe80::/64 orlonger
user@R1# set term 1 from route-filter 2001:db8::/64 orlonger
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

6. Apply the import policy.

```
[edit protocols ripng group ripng-group]
user@R1# set neighbor fe-1/2/0.1 import ripng-import
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        description to-R2;
        family inet6 {
            address 2001:db8:0:1::/64 {
                eui-64;
            }
        }
    }
```

```
}
lo0 {
    unit 1 {
        family inet6 {
            address 2001:db8::1/128;
        }
    }
}
```

```
user@R1# show protocols
ripng {
    group ripng-group {
        export advertise-routes-through-ripng;
        neighbor fe-1/2/0.1 {
            import ripng-import;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-ripng {
    term 1 {
        from protocol [ direct ripng ];
        then accept;
    }
}
policy-statement ripng-import {
    term 1 {
        from {
            route-filter fe80::/64 orlonger;
            route-filter 2001:db8::/64 orlonger;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Looking at the Neighbor Addresses for Device R2

Purpose

Determine the neighbor address that Device R2 is using for Device R1.

Action

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

```
user@R2> show ripng neighbor fe-1/2/0.2

                    Source                          Dest              In
Neighbor     State  Address                         Address  Send Recv Met
--------     -----  -------                         -------  ---- ---- ---
fe-1/2/0.2      Up fe80::2a0:a514:0:24c             ff02::9  yes  yes  1
```

Meaning

Device R2 is using the fe80::2a0:a514:0:24c address to send routes to Device R1.

Looking at the Routes That Device R2 Is Advertising to Device R1

Purpose

Verify that Device R2 is sending the expected routes.

Action

From operational mode, enter the `show route advertising-protocol ripng` command.

```
user@R2> show route advertising-protocol ripng fe80::2a0:a514:0:24c
inet6.0: 17 destinations, 18 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::2/128    *[Direct/0] 3d 22:00:34
                    >    via lo0.2
2001:db8::3/128    *[RIPng/100] 3d 21:47:00, metric 2, tag 0
                    > to fe80::2a0:a514:0:64c via fe-1/2/1.5
2001:db8:0:2::/64  *[Direct/0] 3d 22:00:34
                    >    via fe-1/2/0.2
2001:db8:0:3::/64  *[Direct/0] 3d 22:00:34
                    >    via fe-1/2/1.5
2001:db8:0:4::/64  *[RIPng/100] 3d 21:47:00, metric 2, tag 0
                    > to fe80::2a0:a514:0:64c via fe-1/2/1.5
2002:db7::2/128    *[Direct/0] 00:29:05
                    >    via lo0.2
2002:db8::2/128    *[Direct/0] 00:31:49
                    >    via lo0.2
2002:db9::2/128    *[Direct/0] 00:29:05
                    >    via lo0.2
```

Meaning

Device R2 is sending the extra loopback interface /128 routes to Device R1.

Looking at the Routes That Device R1 Is Receiving from Device R2

Purpose

Verify that Device R1 is receiving the expected routes.

Action

From operational mode, enter the `show route receive-protocol ripng` command.

```
user@R1> show route receive-protocol ripng fe80::2a0:a514:0:24c

inet6.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::2/128    *[RIPng/100] 3d 21:55:49, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8::3/128    *[RIPng/100] 3d 21:55:46, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
```

Meaning

The output shows that the extra loopback interface addresses are excluded.

Checking the Routing Table

Purpose

Verify that the routing table is populated with the expected routes.

Action

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

```
user@R1> show route protocol ripng
inet6.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::2/128    *[RIPng/100] 3d 22:01:40, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
2001:db8::3/128    *[RIPng/100] 3d 22:01:37, metric 3, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
ff02::9/128        *[RIPng/100] 00:00:08, metric 1
                          MultiRecv
```

Meaning

The output shows that the routes have been learned from Device R2 and Device R3.

If you delete or deactivate the import policy, the routing table contains the extra loopback interface routes.

Example: Testing a Routing Policy with Complex Regular Expressions

IN THIS SECTION

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

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

Overview

IN THIS SECTION

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:

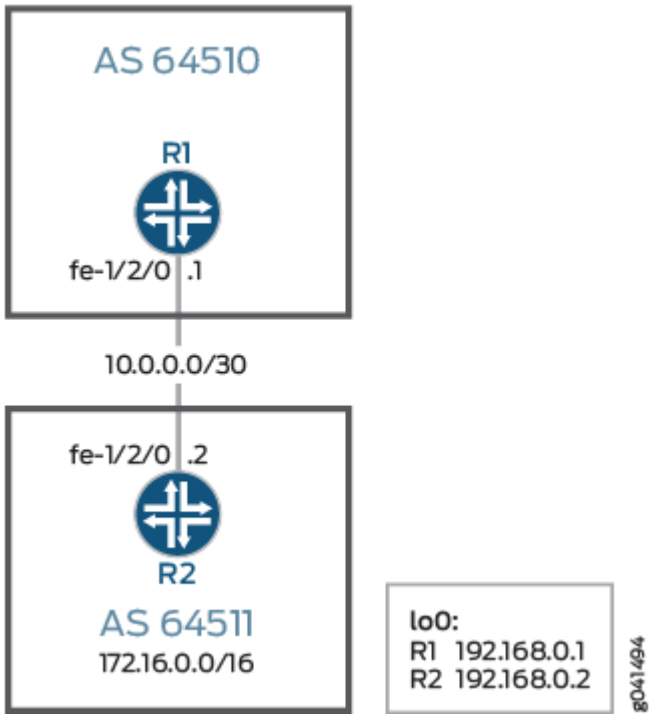
```
policy-statement test-regex {
    term find-routes {
        from community complex-regex;
        then accept;
    }
    term reject-the-rest {
        then reject;
    }
}
community complex-regex members "^64510:[13].*$";
```

This regular expression matches community values beginning with either 1 or 3.

Topology

Figure 20 on page 158 shows the sample network.

Figure 20: Routing Policy Test for Complex Regular Expressions



"CLI Quick Configuration" on page 158 shows the configuration for all of the devices in Figure 20 on page 158.

The section "No Link Title" on page 160 describes the steps on Device R2.

Configuration

IN THIS SECTION

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
```

Procedure

Step-by-Step Procedure

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

To configure 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

IN THIS SECTION

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
- *Understanding How to Define BGP Communities and Extended Communities*
- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions

Traffic Control in a RIPng Network

IN THIS SECTION

Understanding RIPng Traffic Control with Metrics for Optimizing the Path Cost

To tune a RIPng network and to control traffic flowing through the network, you increase or decrease the cost of the paths through the network. RIPng provides two ways to modify the path cost: an incoming metric and an outgoing metric, which are each set to 1 by default. In other words, by default, the metric of routes that RIPng imports from a neighbor or exports to a neighbor is incremented by 1. These routes include those learned from RIPng as well as those learned from other protocols. The metrics are attributes that specify the cost of any route advertised through a host. By increasing or decreasing the metrics—and thus the cost—of links throughout the network, you can control packet transmission across the network.

The incoming metric modifies the cost of an individual segment when a route across the segment is imported into the routing table. For example, if you set the incoming metric on the segment to **3**, the individual segment cost along the link is changed from **1** to **3**. The increased cost affects all route calculations through that link. Other routes that were previously excluded because of a high hop count might now be selected into the router's forwarding table.

The outgoing metric modifies the path cost for all the routes advertised out of a particular interface. Unlike the incoming metric, the outgoing metric modifies the routes that other routers are learning and thereby controls the way they send traffic.

If an exported route was learned from a member of the same RIPng group, the metric associated with that route is the normal RIPng metric. For example, a RIPng route with a metric of 5 learned from a neighbor configured with an incoming metric of 2 is advertised with a combined metric of 7 when advertised to neighbors in the same group. However, if this route was learned from a RIPng neighbor in a different group or from a different protocol, the route is advertised with the metric value configured in the outgoing metric for that group.

You might want to increase the metric of routes to decrease the likelihood that a particular route is selected and installed in the routing table. This process is sometimes referred to as *route poisoning*. Some reasons that you might want to poison a route are that the route is relatively expensive to use, or it has relatively low bandwidth.

A route with a higher metric than another route becomes the active route only when the lower-metric route becomes unavailable. In this way, the higher-metric route serves as a backup path.

One way to increase the metric of imported routes is to configure an import policy. Another way is to include the `metric-in` statement in the RIPng neighbor configuration. One way to increase the metric of export routes is to configure an export policy. Another way is to include the `metric-out` statement in the RIPng neighbor configuration.

Example: Configuring the Metric Value Added to Imported RIPng Routes to Control the Route Selection Process

IN THIS SECTION

This example shows how to change the default metric to be added to incoming routes to control the route selection process.

Requirements

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

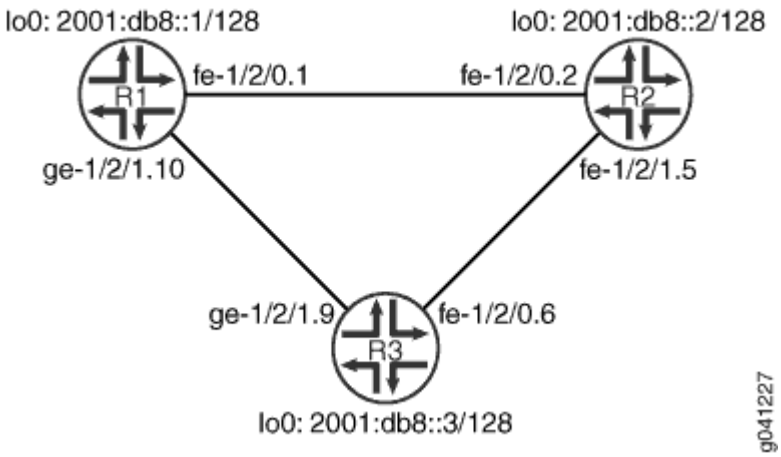
Overview

IN THIS SECTION

Normally, when multiple routes are available, RIPng selects the route with the lowest hop count. Changing the default metric enables you to control the route selection process such that a route with a higher hop count can be preferred over of a route with a lower hop count.

Figure 21 on page 168 shows the topology used in this example.

Figure 21: RIPng Incoming Metrics Network Topology



Device R1 has two potential paths to reach 2001:db8::2/128. The default behavior is to send traffic out the 2001:db8:0:1::/64 interface facing Device R2. Suppose, though, that the path through Device R3 is less expensive to use or has higher bandwidth links. This example shows how to use the `metric-in` statement to ensure that Device R1 uses the path through Device R3 to reach 2001:db8::2/128. "CLI Quick Configuration" on page 168 shows the configuration for all of the devices in Figure 21 on page 168. The section "No Link Title" on page 170 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-1/2/1 unit 10 description to-R3
set interfaces ge-1/2/1 unit 10 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
set protocols ripng group primary export advertise-routes-through-ripng
set protocols ripng group primary neighbor ge-1/2/1.10
set protocols ripng group secondary export advertise-routes-through-ripng
set protocols ripng group secondary neighbor fe-1/2/0.1 metric-in 4
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
set interfaces fe-1/2/1 unit 5 description to-R3
set interfaces fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
set interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.2
set protocols ripng group ripng-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet6 address 2001:db8:0:4::/64 eui-64
set interfaces ge-1/2/1 unit 9 description to-R1
set interfaces ge-1/2/1 unit 9 family inet address 10.0.0.9/30
set interfaces ge-1/2/1 unit 9 family inet6 address 2001:db8:0:6::/64 eui-64
set interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.6
set protocols ripng group ripng-group neighbor ge-1/2/1.9
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
```

```
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

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 a RIPng metrics:

1. Configure the network interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 description to-R2
user@R1# set fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
user@R1# set ge-1/2/1 unit 10 description to-R3
user@R1# set ge-1/2/1 unit 10 family inet6 address 2001:db8:0:5::/64 eui-64
user@R1# set lo0 unit 1 family inet6 address 2001:db8::1/128
```

2. Create the RIPng groups and add the interfaces.

   To configure RIPng in Junos OS, you must configure one or more groups that contain the interfaces on which RIPng is enabled. You do not need to enable RIPng on the loopback interface.

   For the interface that is facing Device R2, the **metric-in 4** setting causes this route to be less likely to be chosen as the active route.

```
[edit protocols ripng]
user@R1# set group primary neighbor ge-1/2/1.10
user@R1# set group secondary neighbor fe-1/2/0.1 metric-in 4
```

3. Create the routing policy to advertise both direct and RIPng-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-ripng term 1]
user@R1# set from protocol direct
user@R1# set from protocol ripng
user@R1# set then accept
```

4. Apply the routing policy.

In Junos OS, you can only apply RIPng export policies at the group level.

```
[edit protocols ripng]
user@R1# set group primary export advertise-routes-through-ripng
user@R1# set group secondary export advertise-routes-through-ripng
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        description to-R2;
        family inet6 {
            address 2001:db8:0:1::/64 {
                eui-64;
            }
        }
    }
}
ge-1/2/1 {
    unit 10 {
        description to-R3;
        family inet6 {
            address 2001:db8:0:5::/64 {
                eui-64;
            }
        }
    }
}
lo0 {
    unit 1 {
        family inet6 {
            address 2001:db8::1/128;
        }
```

```
    }
}
```

```
user@R1# show protocols
ripng {
    group primary {
        export advertise-routes-through-ripng;
        neighbor ge-1/2/0.10;
    }
    group secondary {
        export advertise-routes-through-ripng;
        neighbor fe-1/2/0.1 {
            metric-in 4;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-ripng {
    term 1 {
        from protocol [ direct ripng ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying That the Expected Route Is Active

Purpose

Make sure that Device R1 uses the path through Device R3 to reach 2001:db8:0:2:2a0:a514:0:24c/128.

Action

From operational mode, enter the `show route 2001:db8:0:2:2a0:a514:0:24c` command.

```
user@R1> show route 2001:db8:0:2:2a0:a514:0:24c
inet6.0: 16 destinations, 17 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:0:2::/64  *[RIPng/100] 01:54:35, metric 3, tag 0
                    > to fe80::2a0:a514:0:94c via ge-1/2/1.10
```

Meaning

The **to fe80::2a0:a514:0:94c via ge-1/2/1.10** output shows that Device R1 uses the path through Device R3 to reach 2001:db8:0:2:2a0:a514:0:24c/128. The metric for this route is 3.

Removing the metric-in Statement

Purpose

Delete or deactivate the metric-in statement to see what happens to the 2001:db8:0:2:2a0:a514:0:24c/128 route.

Action

1. From configuration mode, deactivate the `metric-in` statement.

```
[edit protocols ripng group secondary neighbor fe-1/2/0.1]
user@R1# deactivate metric-in
user@R1# commit
```

2. From operational mode, enter the `show route 2001:db8:0:2:2a0:a514:0:24c` command.

```
user@R1> show route 2001:db8:0:2:2a0:a514:0:24c
inet6.0: 16 destinations, 17 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:0:2::/64  *[RIPng/100] 00:00:02, metric 2, tag 0
                    > to fe80::2a0:a514:0:24c via fe-1/2/0.1
```

Meaning

The **to fe80::2a0:a514:0:24c via fe-1/2/0.1** output shows that Device R1 uses the path through Device R2 to reach 2001:db8:0:2:2a0:a514:0:24c/128. The metric for this route is 2.

RIPng Timers

IN THIS SECTION

Example: Configuring RIPng Update Interval

IN THIS SECTION

This example shows how to configure the RIPng update interval and how to monitor the impact of the change.

Requirements

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

Overview

IN THIS SECTION

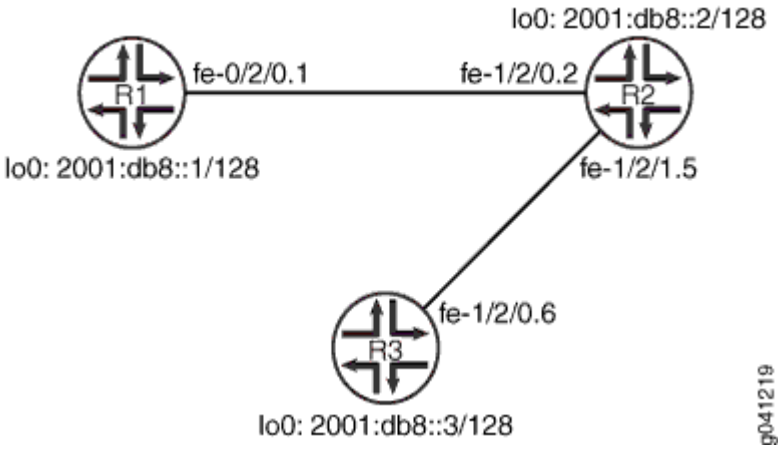
In this example, Device R2 has an update interval of 60 seconds for its neighbor Device R1, and an update interval of 10 seconds for its neighbor Device R3.

This example is not necessarily practical, but it is shown for demonstration purposes. Generally, we recommend against changing the RIPng timers, unless the effects of a change are well understood. Normally, the default values are best left in effect for standard operations.

An export policy is also shown because an export policy is required as part of the minimum configuration for RIPng.

Figure 22 on page 175 shows the topology used in this example.

Figure 22: RIPng Timers Network Topology



"CLI Quick Configuration" on page 176 shows the configuration for all of the devices in Figure 22 on page 175. The section "No Link Title" on page 177 describes the steps on Device R2.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 description to-R1
set interfaces fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
set interfaces fe-1/2/1 unit 5 description to-R3
set interfaces fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
set interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.2 update-interval 60
set protocols ripng group ripng-group neighbor fe-1/2/1.5 update-interval 10
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
```

```
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 description to-R2
set interfaces fe-1/2/0 unit 6 family inet6 address 2001:db8:0:4::/64 eui-64
set interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Step-by-Step Procedure

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

To configure the RIPng update interval:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

   ```
   [edit interfaces]
   user@R2# set fe-1/2/0 unit 2 description to-R1
   user@R2# set fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
   user@R2# set fe-1/2/1 unit 5 description to-R3
   user@R2# set fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
   user@R2# set lo0 unit 2 family inet6 address 2001:db8::2/128
   ```

2. Configure different update intervals for the two RIPng neighbors.

To configure RIPng in Junos OS, you must configure a group that contains the interfaces on which RIPng is enabled. You do not need to enable RIPng on the loopback interface.

```
[edit protocols ripng group ripng-group]
user@R2# set neighbor fe-1/2/0.2 update-interval 60
user@R2# set neighbor fe-1/2/1.5 update-interval 10
```

3. Create the routing policy to advertise both direct and RIPng-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-ripng term 1]
user@R2# set from protocol direct
user@R2# set from protocol ripng
user@R2# set then accept
```

4. Apply the routing policy.

   In Junos OS, you can only apply RIPng export policies at the group level.

```
[edit protocols ripng group ripng-group]
user@R2# set export advertise-routes-through-ripng
```

Results

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

```
user@R2# show interfaces
fe-1/2/0 {
    unit 2 {
        description to-R1;
        family inet6 {
            address 2001:db8:0:2::/64 {
                eui-64;
            }
        }
    }
}
fe-1/2/1 {
```

```
    unit 5 {
        description to-R3;
        family inet6 {
            address 2001:db8:0:3::/64 {
                eui-64;
            }
        }
    }
}
lo0 {
    unit 2 {
        family inet6 {
            address 2001:db8::2/128;
        }
    }
}
```

```
user@R2# show protocols
ripng {
    group ripng-group {
        export advertise-routes-through-ripng;
        neighbor fe-1/2/0.2 {
            update-interval 60;
        }
        neighbor fe-1/2/1.5 {
            update-interval 10;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement advertise-routes-through-ripng {
    term 1 {
        from protocol [ direct ripng ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the RIPng Updates Sent by Device R2

Purpose

Make sure that the RIPng update packets are sent at the expected interval.

Action

From operational mode, enter the `show ripng statistics` command.

```
user@R2> show ripng statistics
RIPng info: port 521; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              4              0              0              0

fe-1/2/0.2:  2 routes learned; 5 routes advertised; timeout 180s; update interval 60s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                        1            1            1
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
Updates Received                    1            0            0
Bad Route Entries                   0            0            0
Updates Ignored                     0            0            0
RIPng Requests Received             0            0            0
RIPng Requests Ignored              0            0            0
```

```
fe-1/2/1.5:  2 routes learned; 5 routes advertised; timeout 180s; update interval 10s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                        6            2            2
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
Updates Received                    2            0            0
Bad Route Entries                   0            0            0
Updates Ignored                     0            0            0
RIPng Requests Received             0            0            0
RIPng Requests Ignored              0            0            0
```

Meaning

The **update interval** field shows that the interval is 60 seconds for its neighbor Device R1 and 10 seconds for its neighbor Device R3. The **Updates Sent** field shows that Device R2 is sending updates to Device R1 at roughly 1/6 of the rate that it is sending updates to Device R3.

Checking the RIPng Updates Received by Device R2

Purpose

Make sure that the RIPng update packets are sent at the expected interval.

Action

From operational mode, enter the `show ripng statistics` command.

```
user@R1> show ripng statistics
RIPng info: port 521; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              8              0              0

fe-1/2/0.1:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                        6            5            2
Triggered Updates Sent              0            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
```

```
Updates Received                  3            3            1
Bad Route Entries                 0            0            0
Updates Ignored                   0            0            0
RIPng Requests Received           0            0            0
RIPng Requests Ignored            0            0            0
```

Meaning

The **Updates Received** field shows the number of updates received from Device R2.

Checking the RIPng Updates Received by Device R3

Purpose

Make sure that the RIPng update packets are sent at the expected interval.

Action

From operational mode, enter the `show ripng statistics` command.

```
user@R3> show ripng statistics
RIPng info: port 521; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              5              0              0              0


fe-1/2/0.6:  5 routes learned; 2 routes advertised; timeout 180s; update interval 30s
Counter                          Total   Last 5 min  Last minute
-------                    -----------  -----------  -----------
Updates Sent                         5            5            2
Triggered Updates Sent               0            0            0
Responses Sent                       0            0            0
Bad Messages                         0            0            0
Updates Received                    16           15            6
Bad Route Entries                    0            0            0
Updates Ignored                      0            0            0
RIPng Requests Received              0            0            0
RIPng Requests Ignored               0            0            0
```

Meaning

The **Updates Received** field shows the number of updates received from Device R2.

SEE ALSO

Understanding RIP Timers | 43

Tracing RIPng Traffic

IN THIS SECTION

Understanding RIPng Protocol Traffic Trace Operations

You can trace various RIPng protocol traffic to help debug RIP protocol issues.

To trace RIP protocol traffic, include the `traceoptions` statement at the `[edit protocols ripng]` hierarchy level:

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

You can specify the following RIPng protocol-specific trace options using the `flag` statement:

- **error**—RIPng error packets
- **expiration**—RIPng route expiration processing
- **holddown**—RIPng hold-down processing

- **nsr-synchronization**—Nonstop routing synchronization events
- **packets**—All RIPng packets
- **request**—RIPng information packets
- **trigger**—RIPng triggered updates
- **update**—RIPng update packets

You can optionally specify one or more of the following flag modifiers:

- **detail**—Detailed trace information
- **receive**—Packets being received
- **send**—Packets being transmitted

NOTE: Use the **detail** flag modifier with caution as this might cause the CPU to become very busy.

Global tracing options are inherited from the configuration set by the `traceoptions` statement at the `[edit routing-options]` hierarchy level. You can override the following global trace options for the RIPng protocol using the `traceoptions flag` statement included at the `[edit protocols ripng]` hierarchy level:

- **all**—All tracing operations
- **general**—All normal operations and routing table changes (a combination of the normal and route trace operations)
- **normal**—Normal events
- **policy**—Policy processing
- **route**—Routing information
- **state**—State transitions
- **task**—Routing protocol task processing
- **timer**—Routing protocol timer processing

NOTE: Use the trace flag **all** with caution as this might cause the CPU to become very busy.

SEE ALSO

Example: Tracing Global Routing Protocol Operations

Example: Tracing RIPng Protocol Traffic

IN THIS SECTION

This example shows how to trace RIPng protocol operations.

Requirements

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

Overview

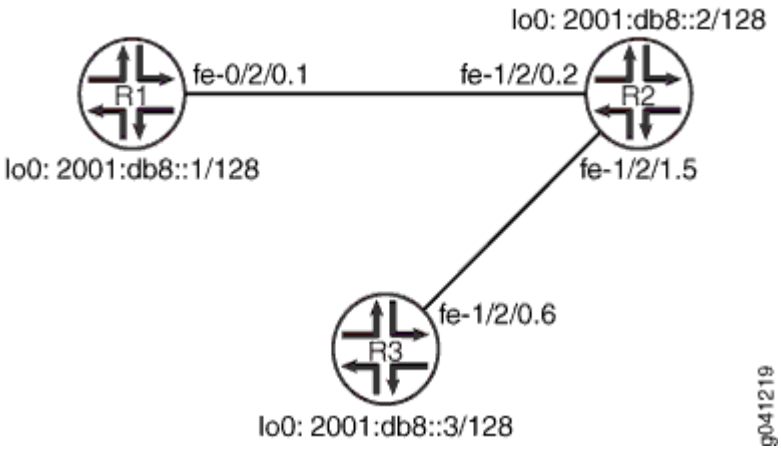
IN THIS SECTION

In this example, Device R1 is set to trace routing information updates.

An export policy is also shown because an export policy is required as part of the minimum configuration for RIPng.

Figure 23 on page 186 shows the topology used in this example.

Figure 23: RIPng Trace Operations Network Topology



"CLI Quick Configuration" on page 186 shows the configuration for all of the devices in Figure 23 on page 186. The section "No Link Title" on page 188 describes the steps on Device R1.

Topology

Configuration

IN THIS SECTION

- Procedure | 186

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces fe-1/2/0 unit 1 description to-R2
set interfaces fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces lo0 unit 1 family inet6 address 2001:db8::1/128
```

```
set protocols ripng traceoptions file ripng-trace-file
set protocols ripng traceoptions flag route
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.1
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R2

```
set interfaces fe-1/2/0 unit 2 description to-R1
set interfaces fe-1/2/0 unit 2 family inet6 address 2001:db8:0:2::/64 eui-64
set interfaces fe-1/2/1 unit 5 description to-R3
set interfaces fe-1/2/1 unit 5 family inet6 address 2001:db8:0:3::/64 eui-64
set interfaces lo0 unit 2 family inet6 address 2001:db8::2/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.2
set protocols ripng group ripng-group neighbor fe-1/2/1.5
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Device R3

```
set interfaces fe-1/2/0 unit 6 description to-R2
set interfaces fe-1/2/0 unit 6 family inet6 address 2001:db8:0:4::/64 eui-64
set interfaces lo0 unit 3 family inet6 address 2001:db8::3/128
set protocols ripng group ripng-group export advertise-routes-through-ripng
set protocols ripng group ripng-group neighbor fe-1/2/0.6
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol direct
set policy-options policy-statement advertise-routes-through-ripng term 1 from protocol ripng
set policy-options policy-statement advertise-routes-through-ripng term 1 then accept
```

Step-by-Step Procedure

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

To configure the RIPng update interval:

1. Configure the network interfaces.

   This example shows multiple loopback interface addresses to simulate attached networks.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 1 description to-R2
user@R1# set fe-1/2/0 unit 1 family inet6 address 2001:db8:0:1::/64 eui-64
user@R1# set lo0 unit 1 family inet6 address 2001:db8::1/128
```

2. Configure the RIPng group, and add the interface to the group.

   To configure RIPng in Junos OS, you must configure a group that contains the interfaces on which RIPng is enabled. You do not need to enable RIPng on the loopback interface.

```
[edit protocols ripng group ripng-group]
user@R1# set neighbor fe-1/2/0.1
```

3. Configure RIPng tracing operations.

```
[edit protocols ripng traceoptions]
user@R1# set file ripng-trace-file
user@R1# set flag route
```

4. Create the routing policy to advertise both direct and RIPng-learned routes.

```
[edit policy-options policy-statement advertise-routes-through-ripng term 1]
user@R1# set from protocol direct
user@R1# set from protocol ripng
user@R1# set then accept
```

5. Apply the routing policy.

   In Junos OS, you can only apply RIPng export policies at the group level.

```
[edit protocols ripng group ripng-group]
user@R1# set export advertise-routes-through-ripng
```

Results

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

```
user@R1# show interfaces
fe-1/2/0 {
    unit 1 {
        description to-R2;
        family inet6 {
            address 2001:db8:0:1::/64 {
                eui-64;
            }
        }
    }
}
lo0 {
    unit 1 {
        family inet6 {
            address 2001:db8::1/128;
        }
    }
}
```

```
user@R1# show protocols
ripng {
    traceoptions {
        file ripng-trace-file;
        flag route;
    }
    group ripng-group {
        export advertise-routes-through-ripng;
        neighbor fe-1/2/0.1;
    }
}
```

```
user@R1# show policy-options
policy-statement advertise-routes-through-ripng {
```

```
    term 1 {
        from protocol [ direct ripng ];
        then accept;
    }
}
```

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

Verification

IN THIS SECTION

- Checking the Log File | **190**

Confirm that the configuration is working properly.

Checking the Log File

Purpose

Make sure that the RIPng route updates are logged in the configured log file.

Action

1. Deactivate the extra loopback interface address on Device R3.

```
[edit  interfaces lo0 unit 3 family inet6]
user@R3# deactivate address 2001:db8::3/128
user@R3# commit
```

2. From operational mode, enter the `show log ripng-trace-file` command with the **| match 2001:db8::3** option.

```
user@R1> show log ripng-trace-file | match 2001:db8::3

Mar  6 14:57:03.516867  2001:db8::3/128: metric-in: 3, change: 3 -> 3; # gw: 1, pkt_upd_src
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:57:32.786286  2001:db8::3/128: metric-in: 3, change: 3 -> 3; # gw: 1, pkt_upd_src
```

```
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:58:02.584669  2001:db8::3/128: metric-in: 3, change: 3 -> 3; # gw: 1, pkt_upd_src
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:58:30.213894  2001:db8::3/128: metric-in: 3, change: 3 -> 3; # gw: 1, pkt_upd_src
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:59:00.115110  2001:db8::3/128: metric-in: 3, change: 3 -> 3; # gw: 1, pkt_upd_src
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:59:05.826644 Setting RIPng rtbit on route 2001:db8::3/128, tsi = 0xbb69880
Mar  6 14:59:13.014652  2001:db8::3/128: metric-in: 16, change: 3 -> 16; # gw: 1, pkt_upd_src
fe80::2a0:a514:0:24c, inx: 0, rte_upd_src fe80::2a0:a514:0:24c
Mar  6 14:59:13.015132 CHANGE   2001:db8::3/128     nhid 566 gw fe80::2a0:a514:0:24c RIPng
pref 100/0 metric 3/0 fe-1/2/0.1 **Delete Int>
Mar  6 14:59:13.015197 Best route to 2001:db8::3/128 got deleted. Doing route calculation on
the stored rte-info
```

Meaning

The output shows that the route to 2001:db8::3/128 was deleted.

4

CHAPTER

Troubleshooting

IN THIS CHAPTER

Troubleshooting Network Issues

IN THIS SECTION

Working with Problems on Your Network

IN THIS SECTION

Problem

Description

This checklist provides links to troubleshooting basics, an example network, and includes a summary of the commands you might use to diagnose problems with the router and network.

Solution

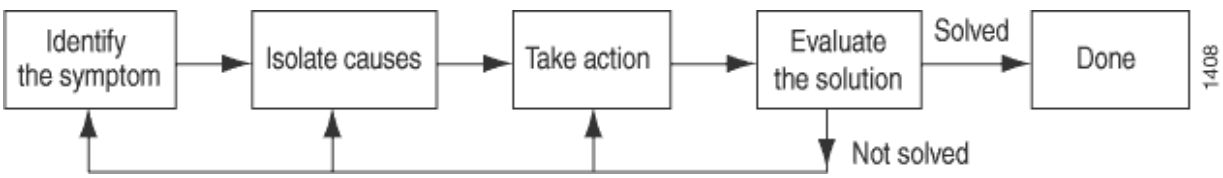
Table 3: Checklist for Working with Problems on Your Network

Tasks	Command or Action
Isolating a Broken Network Connection	
1. *Identifying the Symptoms of a Broken Network Connection*	**ping (*ip-address* | *hostname*) show route (*ip-address* | *hostname*) traceroute (*ip-address* | *hostname*)**
1. *Isolating the Causes of a Network Problem*	`show < configuration | interfaces | protocols | route >`
1. *Taking Appropriate Action for Resolving the Network Problem*	`[edit] delete routing options static route` *`destination-prefix`* **commit and-quit show route *destination-prefix***
1. *Evaluating the Solution to Check Whether the Network Problem Is Resolved*	`show route (`*`ip-address`* `|` *`hostname`*`) ping (`*`ip-address`* `|` *`hostname`*`)` **count 3 traceroute (*ip-address* | *hostname*)**

Isolating a Broken Network Connection

By applying the standard four-step process illustrated in Figure 24 on page 194, you can isolate a failed node in the network. Note that the functionality described in this section is not supported in versions 15.1X49, 15.1X49-D30, or 15.1X49-D40.

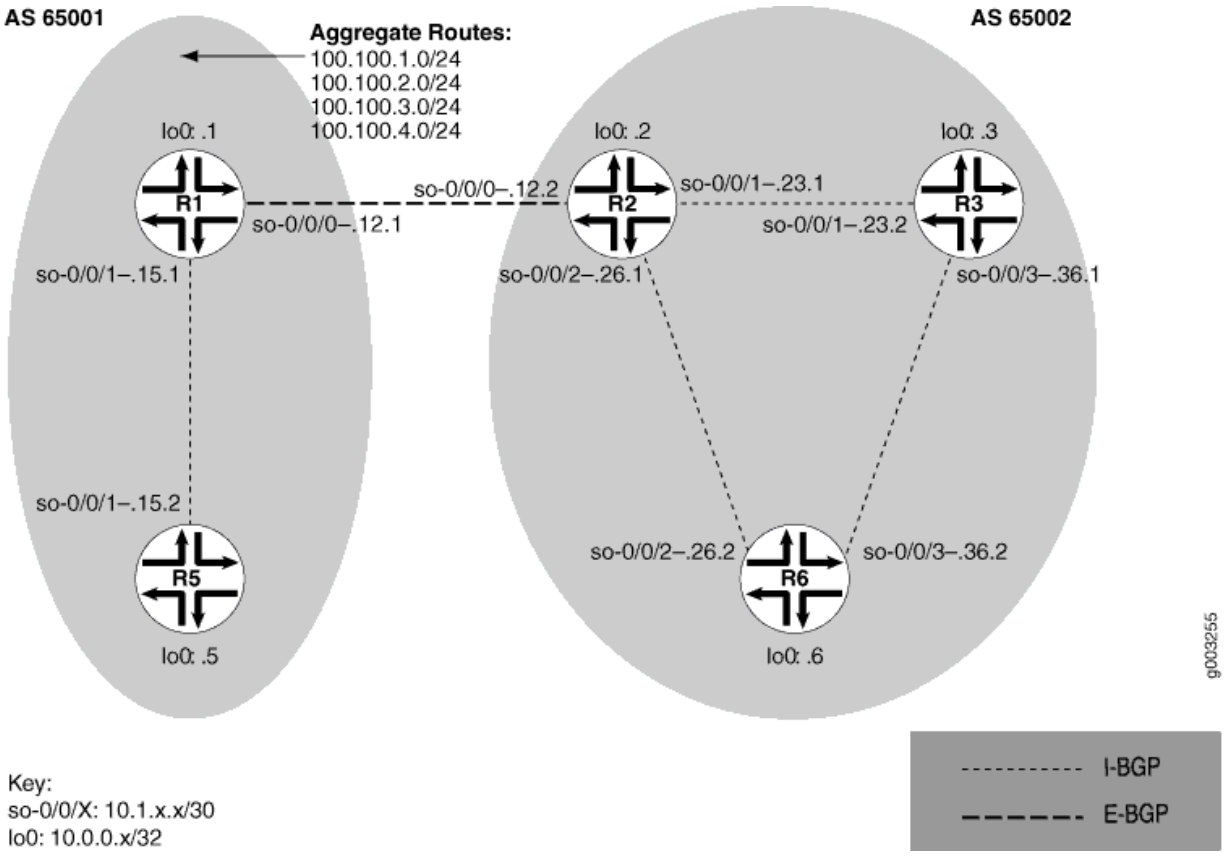
Figure 24: Process for Diagnosing Problems in Your Network



Before you embark on the four-step process, however, it is important that you are prepared for the inevitable problems that occur on all networks. While you might find a solution to a problem by simply trying a variety of actions, you can reach an appropriate solution more quickly if you are systematic in your approach to the maintenance and monitoring of your network. To prepare for problems on your network, understand how the network functions under normal conditions, have records of baseline network activity, and carefully observe the behavior of your network during a problem situation.

Figure 25 on page 195 shows the network topology used in this topic to illustrate the process of diagnosing problems in a network.

Figure 25: Network with a Problem



The network in Figure 25 on page 195 consists of two autonomous systems (ASs). AS 65001 includes two routers, and AS 65002 includes three routers. The border router (`R1`) in AS 65001 announces aggregated prefixes `100.100/24` to the AS 65002 network. The problem in this network is that `R6` does not have access to `R5` because of a loop between `R2` and `R6`.

To isolate a failed connection in your network, follow the steps in these topics:

- *Isolating the Causes of a Network Problem*

- *Taking Appropriate Action for Resolving the Network Problem*
- *Taking Appropriate Action for Resolving the Network Problem*
- *Evaluating the Solution to Check Whether the Network Problem Is Resolved*

Identifying the Symptoms of a Broken Network Connection

IN THIS SECTION

Problem

Description

The symptoms of a problem in your network are usually quite obvious, such as the failure to reach a remote host.

Solution

To identify the symptoms of a problem on your network, start at one end of your network and follow the routes to the other end, entering all or one of the following Junos OS command-line interfaces (CLI) operational mode commands:

```
            user@host> ping (ip-address | host-name)
user@host> show route (ip-address | host-name)
user@host> traceroute (ip-address | host-name)
```

Sample Output

```
user@R6> ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5): 56 data bytes
36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
```

```
 4  5  00 0054 e2db   0 0000  01  01 a8c6 10.1.26.2  10.0.0.5

36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 e2de   0 0000  01  01 a8c3 10.1.26.2  10.0.0.5

36 bytes from 10.1.26.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 e2e2   0 0000  01  01 a8bf 10.1.26.2  10.0.0.5

^C
--- 10.0.0.5 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss

user@R6> show route 10.0.0.5

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[IS-IS/165] 00:02:39, metric 10
                     > to 10.1.26.1  via so-0/0/2.0

user@R6> traceroute 10.0.0.5
traceroute to 10.0.0.5 (10.0.0.5), 30 hops max, 40 byte packets
 1  10.1.26.1 (10.1.26.1)  0.649 ms  0.521 ms  0.490 ms
 2  10.1.26.2 (10.1.26.2)  0.521 ms  0.537 ms  0.507 ms
 3  10.1.26.1 (10.1.26.1)  0.523 ms  0.536 ms  0.514 ms
 4  10.1.26.2 (10.1.26.2)  0.528 ms  0.551 ms  0.523 ms
 5  10.1.26.1 (10.1.26.1)  0.531 ms  0.550 ms  0.524 ms
```

Meaning

The sample output shows an unsuccessful `ping` command in which the packets are being rejected because the time to live is exceeded. The output for the `show route` command shows the interface (`10.1.26.1`) that you can examine further for possible problems. The `traceroute` command shows the loop between `10.1.26.1` (R2) and `10.1.26.2` (R6), as indicated by the continuous repetition of the two interface addresses.

Isolating the Causes of a Network Problem

IN THIS SECTION

Problem

Description

A particular symptom can be the result of one or more causes. Narrow down the focus of your search to find each individual cause of the unwanted behavior.

Solution

To isolate the cause of a particular problem, enter one or all of the following Junos OS CLI operational mode command:

```
user@host> show < configuration | bgp | interfaces | isis | ospf | route
>
```

Your particular problem may require the use of more than just the commands listed above. See the appropriate command reference for a more exhaustive list of commonly used operational mode commands.

Sample Output

```
user@R6> show interfaces terse
Interface               Admin Link Proto Local                 Remote
so-0/0/0                up    up
so-0/0/0.0              up    up   inet  10.1.56.2/30
                                   iso
so-0/0/2                up    up
so-0/0/2.0              up    up   inet  10.1.26.2/30
                                   iso
so-0/0/3                up    up
```

```
so-0/0/3.0              up    up   inet  10.1.36.2/30
                                   iso
[...Output truncated...]
```

The following sample output is from R2:

```
user@R2> show route 10.0.0.5

inet.0: 22 destinations, 25 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32         *[Static/5] 00:16:21
                    > to 10.1.26.2 via so-0/0/2.0
                    [BGP/170] 3d 20:23:35, MED 5, localpref 100
                      AS path: 65001 I
                    > to 10.1.12.1 via so-0/0/0.0
```

Meaning

The sample output shows that all interfaces on R6 are up. The output from R2 shows that a static route [Static/5] configured on R2 points to R6 (10.1.26.2) and is the preferred route to R5 because of its low preference value. However, the route is looping from R2 to R6, as indicated by the missing reference to R5 (10.1.15.2).

Taking Appropriate Action for Resolving the Network Problem

IN THIS SECTION

Problem

Description

The appropriate action depends on the type of problem you have isolated. In this example, a static route configured on R2 is deleted from the [routing-options] hierarchy level. Other appropriate actions might include the following:

Solution

- Check the local router's configuration and edit it if appropriate.
- Troubleshoot the intermediate router.
- Check the remote host configuration and edit it if appropriate.
- Troubleshoot routing protocols.
- Identify additional possible causes.

To resolve the problem in this example, enter the following Junos OS CLI commands:

```
[edit]
                          user@R2# delete routing-options static route destination-
prefix
user@R2# commit and-quit
user@R2# show route destination-prefix
```

Sample Output

```
[edit]
user@R2# delete routing-options static route 10.0.0.5/32

[edit]
user@R2# commit and-quit
commit complete
Exiting configuration mode

user@R2> show route 10.0.0.5

inet.0: 22 destinations, 24 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
10.0.0.5/32        *[BGP/170] 3d 20:26:17, MED 5, localpref 100
                      AS path: 65001 I
                    > to 10.1.12.1 via so-0/0/0.0
```

Meaning

The sample output shows the static route deleted from the [routing-options] hierarchy and the new configuration committed. The output for the show route command now shows the BGP route as the preferred route, as indicated by the asterisk (*).

Evaluating the Solution to Check Whether the Network Problem Is Resolved

IN THIS SECTION

Problem

Description

If the problem is solved, you are finished. If the problem remains or a new problem is identified, start the process over again.

You can address possible causes in any order. In relation to the network in *Isolating a Broken Network Connection*, we chose to work from the local router toward the remote router, but you might start at a different point, particularly if you have reason to believe that the problem is related to a known issue, such as a recent change in configuration.

Solution

To evaluate the solution, enter the following Junos OS CLI commands:

```
                                    user@host> show route (ip-address   |host-name)
user@host> ping (ip-address  | host-name)
user@host> traceroute (ip-address  | host-name)
```

Sample Output

```
user@R6> show route 10.0.0.5

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32         *[BGP/170]  00:01:35, MED 5, localpref 100, from 10.0.0.2
                      AS path: 65001 I
                    > to 10.1.26.1 via so-0/0/2.0

user@R6> ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5): 56 data bytes
64 bytes from 10.0.0.5: icmp_seq=0 ttl=253 time=0.866 ms
64 bytes from 10.0.0.5: icmp_seq=1 ttl=253 time=0.837 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=253 time=0.796 ms
^C
--- 10.0.0.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.796/0.833/0.866/0.029 ms

user@R6> traceroute 10.0.0.5
traceroute to 10.0.0.5 (10.0.0.5), 30 hops max, 40 byte packets
 1  10.1.26.1 (10.1.26.1)  0.629 ms  0.538 ms  0.497 ms
 2  10.1.12.1 (10.1.12.1)  0.534 ms  0.538 ms  0.510 ms
 3  10.0.0.5 (10.0.0.5)  0.776 ms  0.705 ms  0.672 ms
```

Meaning

The sample output shows that there is now a connection between R6 and R5. The `show route` command shows that the BGP route to R5 is preferred, as indicated by the asterisk (*). The `ping` command is successful and the `traceroute` command shows that the path from R6 to R5 is through R2 (10.1.26.1), and then through R1 (10.1.12.1).

Checklist for Tracking Error Conditions

IN THIS SECTION

Problem

Description

Table 4 on page 203 provides links and commands for configuring routing protocol daemon tracing, Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS) protocol, and Open Shortest Path First (OSPF) protocol tracing to diagnose error conditions.

Solution

Table 4: Checklist for Tracking Error Conditions

Tasks	Command or Action
Configure Routing Protocol Process Tracing	
1. *Configure Routing Protocol Process Tracing*	`[edit] edit routing-options traceoptions` *`filename`* `size` *`size`* **files** ***number*** **show co** **log** ***filename***
1. *Configure Routing Protocol Tracing for a Specific Routing Protocol*	`[edit] edit protocol` *`protocol-name`* **trace** ***filename*** **size** ***size*** **files** ***number*** **show con** **log** ***filename***
1. *Monitor Trace File Messages Written in Near-Real Time*	**monitor start** ***filename***
1. *Stop Trace File Monitoring*	**monitor stop** ***filename***

Table 4: Checklist for Tracking Error Conditions *(Continued)*

Tasks	Command or Action
Configure BGP-Specific Options	
1. Display Detailed BGP Protocol Information	[edit] edit protocol bgp traceoptions se detail show commit run show log *filename*
1. Display Sent or Received BGP Packets	[edit] edit protocol bgp traceoptions se (send | receive) show commit run show lo
1. Diagnose BGP Session Establishment Problems	[edit] edit protocol bgp set traceoptior detail show commit run show log *filename*
Configure IS-IS-Specific Options	
1. Displaying Detailed IS-IS Protocol Information	[edit] edit protocol isis traceoptions s detail show commit run show log *filename*
1. Displaying Sent or Received IS-IS Protocol Packets	[edit] edit protocols isis traceoptions (send | receive) show commit run show lo
1. Analyzing IS-IS Link-State PDUs in Detail	[edit] edit protocols isis traceoptions detail show commit run show log *filename*
Configure OSPF-Specific Options	
1. Diagnose OSPF Session Establishment Problems	[edit] edit protocols ospf traceoptions detail show commit run show log *filename*
1. Analyze OSPF Link-State Advertisement Packets in Detail	[edit] edit protocols ospf traceoptions update detail show commit run show log

Configure Routing Protocol Process Tracing

IN THIS SECTION

Action

To configure routing protocol process (rpd) tracing, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
user@host# edit routing-options traceoptions
```

2. Configure the file, file size, number, and flags:

```
[edit routing-options traceoptions]
user@host#  set file filename size size file number
[edit routing-options traceoptions]
user@host#  set flag flag
```

For example:

```
[edit routing-options traceoptions]
user@host# set file daemonlog size 10240 files 10
[edit routing-options traceoptions]
user@host# set flag general
```

3. Verify the configuration:

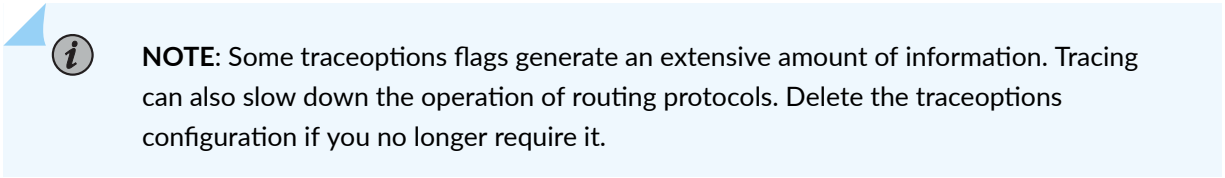
```
user@host# show
```

For example:

```
[edit routing-options traceoptions]
user@host# show
file daemonlog size 10k files 10;
flag general;
```

4. Commit the configuration:

```
user@host# commit
```



NOTE: Some traceoptions flags generate an extensive amount of information. Tracing can also slow down the operation of routing protocols. Delete the traceoptions configuration if you no longer require it.

1. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit routing-options traceoptions]
user@pro4-a# run show log daemonlog
Sep 17 14:17:31 trace_on: Tracing to "/var/log/daemonlog" started
Sep 17 14:17:31 Tracing flags enabled: general
Sep 17 14:17:31 inet_routerid_notify: Router ID: 10.255.245.44
Sep 17 14:17:31 inet_routerid_notify: No Router ID assigned
Sep 17 14:17:31 Initializing LSI globals
Sep 17 14:17:31 LSI initialization complete
Sep 17 14:17:31 Initializing OSPF instances
Sep 17 14:17:31 Reinitializing OSPFv2 instance master
Sep 17 14:17:31 OSPFv2 instance master running
[...Output truncated...]
```

Meaning

Table 5 on page 207 lists tracing flags and example output for Junos-supported routing protocol daemon tracing.

Table 5: Routing Protocol Daemon Tracing Flags

Tracing Flag	Description	Example Output
all	All operations	Not available.
general	Normal operations and routing table change	Not available.
normal	Normal operations	Not available.
policy	Policy operations and actions	**Nov 29 22:19:58 export: Dest 10.0.0.0 proto Static Nov 29 22:19:58 policy_match_qual_or: Qualifier proto Sense: 0 Nov 29 22:19:58 policy_match_qual_or: Qualifier proto Sense: 0 Nov 29 22:19:58 export: Dest 10.10.10.0 proto IS-IS**
route	Routing table changes	**Nov 29 22:23:59 Nov 29 22:23:59 rtlist_walker_job: rt_list walk for RIB inet.0 started with 42 entries Nov 29 22:23:59 rt_flash_update_callback: flash KRT (inet.0) start Nov 29 22:23:59 rt_flash_update_callback: flash KRT (inet.0) done Nov 29 22:23:59 rtlist_walker_job: rt_list walk for inet.0 ended with 42 entries Nov 29 22:23:59 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 CHANGE route/user af 2 addr 172.16.0.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 ADD route/user af 2 addr 172.17.0.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:23:59 KRT Request: send len 68 v14 seq 0 ADD route/user af 2 addr 10.149.3.0 nhop-type unicast nhop 10.10.10.33 Nov 29 22:24:19 trace_on: Tracing to "/var/log/rpdlog" started Nov 29 22:24:19 KRT Request: send len 68 v14 seq 0 DELETE route/user af 2 addr 10.10.218.0 nhop-type unicast nhop 10.10.10.29 Nov 29 22:24:19 RELEASE 10.10.218.0 255.255.255.0 gw 10.10.10.29,10.10.10.33 BGP pref 170/-101 metric so-1/1/0.0,so-1/1/1.0 <Release Delete Int Ext> as 65401 Nov 29 22:24:19 KRT Request: send len 68 v14 seq 0 DELETE route/user af 2 addr 172.18.0.0 nhop-type unicast nhop 10.10.10.33**
state	State transitions	Not available.

Table 5: Routing Protocol Daemon Tracing Flags *(Continued)*

Tracing Flag	Description	Example Output
task	Interface transactions and processing	**Nov 29 22:50:04 foreground dispatch running job task_collect for task Scheduler Nov 29 22:50:04 task_collect_job: freeing task MGMT_Listen (DELETED) Nov 29 22:50:04 foreground dispatch completed job task_collect for task Scheduler Nov 29 22:50:04 background dispatch running job rt_static_update for task RT Nov 29 22:50:04 task_job_delete: delete background job rt_static_update for task RT Nov 29 22:50:04 background dispatch completed job rt_static_update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 background dispatch returned job Flash update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 task_job_delete: delete background job Flash update for task RT Nov 29 22:50:04 background dispatch completed job Flash update for task RT Nov 29 22:50:04 background dispatch running job Flash update for task RT Nov 29 22:50:04 task_job_delete: delete background job Flash update for task RT**
timer	Timer usage	**Nov 29 22:52:07 task_timer_hiprio_dispatch: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 1 timer Nov 29 22:52:07 task_timer_hiprio_dispatch: running high priority timer queue Nov 29 22:52:07 task_timer_hiprio_dispatch: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 1 timer Nov 29 22:52:07 main: running normal priority timer queue Nov 29 22:52:07 main: ran 2 timers**

Configure Routing Protocol Tracing for a Specific Routing Protocol

IN THIS SECTION

Action

To configure routing protocol tracing for a specific routing protocol, follow these steps:

1. In configuration mode, go to the following hierarchy level:

```
[edit]
                user@host# edit protocol protocol-name traceoptions
```

2. Configure the file, file size, number, and flags:

```
[edit protocols protocol name traceoptions]
                user@host# set file filename size size files
number
[edit protocols protocol name traceoptions]
user@host# set flag flag
```

   For example:

```
[edit protocols ospf traceoptions]
user@host# set file ospflog size 10240 files 10
[edit protocols ospf traceoptions]
user@host# set flag general
```

3. Verify the configuration:

```
user@host# show
```

   For example:

```
[edit protocols ospf traceoptions]
user@host# show
file ospflog size 10k files 10;
flag general;
```

4. Commit the configuration:

```
user@host# commit
```

5. View the contents of the file containing the detailed messages:

```
user@host# run show log filename
```

For example:

```
[edit protocols ospf traceoptions]
user@pro4-a# run show log ospflog
Sep 17 14:23:10 trace_on: Tracing to "/var/log/ospflog" started
Sep 17 14:23:10 rt_flash_update_callback: flash OSPF (inet.0) start
Sep 17 14:23:10 OSPF:  multicast address 224.0.0.5/32, route ignored
Sep 17 14:23:10 rt_flash_update_callback: flash OSPF (inet.0) done
Sep 17 14:23:10 CHANGE   10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Delete Int>
Sep 17 14:23:10 CHANGE   10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 ADD      10.255.245.46/32   gw 10.10.208.67  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 CHANGE   10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Delete Int>
Sep 17 14:23:10 CHANGE   10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 ADD      10.255.245.48/32   gw 10.10.208.69  OSPF     pref 10/0 metric 1/0
fe-0/0/0.0 <Active Int>
Sep 17 14:23:10 rt_close: 4/4 routes proto OSPF
[...Output truncated...]
```

Meaning

Table 6 on page 210 lists standard tracing options that are available globally or that can be applied to specific protocols. You can also configure tracing for a specific BGP peer or peer group. For more information, see the *Junos System Basics Configuration Guide*.

Table 6: Standard Trace Options for Routing Protocols

Tracing Flag	Description
all	All operations

Table 6: Standard Trace Options for Routing Protocols *(Continued)*

Tracing Flag	Description
general	Normal operations and routing table changes
normal	Normal operations
policy	Policy operations and actions
route	Routing table changes
state	State transitions
task	Interface transactions and processing
timer	Timer usage

Monitor Trace File Messages Written in Near-Real Time

IN THIS SECTION

Purpose

To monitor messages in near-real time as they are being written to a trace file.

Action

To monitor messages in near-real time as they are being written to a trace file, use the following Junos OS command-line interface (CLI) operational mode command:

```
user@host> monitor start  filename
```

Sample Output

command-name

```
user@host> monitor start isis
user@host>
*** isis ***
Sep 15 18:32:21 Updating LSP isis5.02-00 in database
Sep 15 18:32:21 Updating L2 LSP isis5.02-00 in TED
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Scheduling L2 LSP isis5.02-00 sequence 0xd87 on interface fxp2.3
Sep 15 18:32:21 Updating LSP isis5.00-00 in database
Sep 15 18:32:21 Updating L1 LSP isis5.00-00 in TED
Sep 15 18:32:21 Sending L2 LSP isis5.02-00 on interface fxp2.3
Sep 15 18:32:21     sequence 0xd87, checksum 0xc1c8, lifetime 1200
```

Stop Trace File Monitoring

IN THIS SECTION

Action

To stop monitoring a trace file in near-real time, use the following Junos OS CLI operational mode command after you have started monitoring:

```
user@host              monitor stop filename
```

Sample Output

```
user@host> monitor start isis
user@host>
*** isis ***
Sep 15 18:32:21 Updating LSP isis5.02-00 in database
Sep 15 18:32:21 Updating L2 LSP isis5.02-00 in TED
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis6.00
Sep 15 18:32:21 Adding a half link from isis5.02 to isis5.00
Sep 15 18:32:21 Scheduling L2 LSP isis5.02-00 sequence 0xd87 on interface fxp2.3
Sep 15 18:32:21 Updating LSP isis5.00-00 in database
Sep 15 18:32:21 Updating L1 LSP isis5.00-00 in TED
Sep 15 18:32:21 Sending L2 LSP isis5.02-00 on interface fxp2.3
Sep 15 18:32:21     sequence 0xd87, checksum 0xc1c8, lifetime 1200
monitor stop isis
user@host>
```

Monitoring RIP Traffic

IN THIS SECTION

Monitoring RIP Routing Information

IN THIS SECTION

Purpose

NOTE: This topic applies only to the J-Web Application package.

Use the monitoring functionality to monitor RIP routing on routing devices.

Action

To view RIP routing information in the J-Web interface, select **Monitor** > **Routing** > **RIP Information**.

To view RIP routing information in the CLI, enter the following CLI commands:

- `show rip statistics`
- `show rip neighbor`

Meaning

Table 7 on page 214 summarizes key output fields in the RIP routing display in the J-Web interface.

Table 7: Summary of Key RIP Routing Output Fields

Field	Values	Additional Information
RIP Statistics		
Protocol Name	The RIP protocol name.	

Table 7: Summary of Key RIP Routing Output Fields *(Continued)*

Field	Values	Additional Information
Port number	The port on which RIP is enabled.	
Hold down time	The interval during which routes are neither advertised nor updated.	
Global routes learned	Number of RIP routes learned on the logical interface.	
Global routes held down	Number of RIP routes that are not advertised or updated during the hold-down interval.	
Global request dropped	Number of requests dropped.	
Global responses dropped	Number of responses dropped.	
RIP Neighbors		
Neighbor	Name of the RIP neighbor.	This value is the name of the interface on which RIP is enabled. Click the name to see the details for this neighbor.
State	State of the RIP connection: **Up** or **Dn** (Down).	
Source Address	Local source address.	This value is the configured address of the interface on which RIP is enabled.
Destination Address	Destination address.	This value is the configured address of the immediate RIP adjacency.
Send Mode	The mode of sending RIP messages.	

Table 7: Summary of Key RIP Routing Output Fields *(Continued)*

Field	Values	Additional Information
Receive Mode	The mode in which messages are received.	
In Metric	Value of the incoming metric configured for the RIP neighbor.	

Verifying a RIP Configuration

IN THIS SECTION

To verify a RIP configuration, perform the following tasks:

Verifying the RIP-Enabled Interfaces

IN THIS SECTION

Purpose

Verify that all the RIP-enabled interfaces are available and active.

Action

From the CLI, enter the `show rip neighbor` command.

Sample Output

command-name

```
user@host> show rip neighbor
Source          Destination     Send   Receive   In
Neighbor          State  Address         Address         Mode   Mode     Met
--------          -----  -------         -----------     ----   -------  ---
ge-0/0/0.0           Dn (null)          (null)          mcast  both       1
ge-0/0/1.0           Up 192.168.220.5   224.0.0.9       mcast  both       1
```

Meaning

The output shows a list of the RIP neighbors that are configured on the device. Verify the following information:

- Each configured interface is present. Interfaces are listed in alphabetical order.
- Each configured interface is up. The state of the interface is listed in the **Destination State** column. A state of **Up** indicates that the link is passing RIP traffic. A state of **Dn** indicates that the link is not passing RIP traffic. In a point-to-point link, this state generally means that either the end point is not configured for RIP or the link is unavailable.

Verifying Reachability of All Hosts in the RIP Network

IN THIS SECTION

Purpose

By using the traceroute tool on each loopback address in the network, verify that all hosts in the RIP network are reachable from each Juniper Networks device.

Action

For each device in the RIP network:

1. In the J-Web interface, select **Troubleshoot>Traceroute**.
2. In the Remote Host box, type the name of a host for which you want to verify reachability from the device.
3. Click **Start**. Output appears on a separate page.

Sample Output

command-name

```
1 172.17.40.254 (172.17.40.254) 0.362 ms 0.284 ms 0.251 ms
2 routera-fxp0.englab.mycompany.net (192.168.71.246) 0.251 ms 0.235 ms 0.200 ms
```

Meaning

Each numbered row in the output indicates a routing hop in the path to the host. The three-time increments indicate the round-trip time (RTT) between the device and the hop for each traceroute packet.

To ensure that the RIP network is healthy, verify the following information:

- The final hop in the list is the host you want to reach.
- The number of expected hops to the host matches the number of hops in the traceroute output. The appearance of more hops than expected in the output indicates that a network segment is probably unreachable. It might also indicate that the incoming or outgoing metric on one or more hosts has been set unexpectedly.

Verifying the Exchange of RIP Messages

IN THIS SECTION

Purpose

Verify that RIP messages are being sent and received on all RIP-enabled interfaces.

Action

From the CLI, enter the `show rip statistics` command.

Sample Output

command-name

```
user@host> show rip statistics
RIPv2 info: port 520; holddown 120s.
    rts learned  rts held down  rqsts dropped  resps dropped
              10              0              0              0

t1-0/0/2.0:  0 routes learned; 13 routes advertised; timeout 120s; update interval 45s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                     2855           11            2
Triggered Updates Sent              5            0            0
Responses Sent                      0            0            0
Bad Messages                        0            0            0
RIPv1 Updates Received              0            0            0
RIPv1 Bad Route Entries             0            0            0
RIPv1 Updates Ignored               0            0            0
RIPv2 Updates Received             41            0            0
RIPv2 Bad Route Entries             0            0            0
```

```
RIPv2 Updates Ignored               0            0            0
Authentication Failures             0            0            0
RIP Requests Received               0            0            0
RIP Requests Ignored                0            0            0

ge-0/0/1.0:  10 routes learned; 3 routes advertised; timeout 180s; update interval 30s
Counter                         Total   Last 5 min  Last minute
-------                   -----------  -----------  -----------
Updates Sent                     2855           11            2
Triggered Updates Sent              3            0            0
Responses Sent                      0            0            0
Bad Messages                        1            0            0
RIPv1 Updates Received              0            0            0
RIPv1 Bad Route Entries             0            0            0
RIPv1 Updates Ignored               0            0            0
RIPv2 Updates Received           2864           11            2
RIPv2 Bad Route Entries            14            0            0
RIPv2 Updates Ignored               0            0            0
Authentication Failures             0            0            0
RIP Requests Received               0            0            0
RIP Requests Ignored                0            0            0
```

Meaning

The output shows the number of RIP routes learned. It also shows the number of RIP updates sent and received on the RIP-enabled interfaces. Verify the following information:

- The number of RIP routes learned matches the number of expected routes learned. Subnets learned by direct connectivity through an outgoing interface are not listed as RIP routes.
- RIP updates are being sent on each RIP-enabled interface. If no updates are being sent, the routing policy might not be configured to export routes.
- RIP updates are being received on each RIP-enabled interface. If no updates are being received, the routing policy might not be configured to export routes on the host connected to that subnet. The lack of updates might also indicate an authentication error.

RELATED DOCUMENTATION

RIP Configuration Overview | 12

show rip statistics

CLI Explorer

show rip neighbor

CLI Explorer

traceroute

CLI Explorer

RIP Overview | **2**

Example: Configuring the Sending and Receiving of RIPv1 and RIPv2 Packets | **117**

5

CHAPTER

Configuration Statements and Operational Commands

IN THIS CHAPTER

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Read this guide to learn about the syntax and options that make up the statements and commands. Also understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands