

# Chapter 8

## Configure Protocol-Independent Routing Properties

This chapter describes the following tasks for configuring and monitoring routing properties:

- Create IPv6 Routing Tables on page 74
- Configure Static Routes on page 75
- Configure Martian Addresses on page 84
- Create Routing Table Groups on page 85
- Configure How Interface Routes Are Imported into Routing Tables on page 86
- Configure a Route Limit for Routing Tables on page 86
- Install a Static Route into More than One Routing Table on page 87
- Configure the AS Number on page 87
- Configure the Router Identifier on page 88
- Configure Graceful Restart on page 88
- Configure AS Confederation Members on page 88
- Configure Logging for the Routing Protocol Process on page 89
- Trace Global Routing Protocol Operations on page 90
- Route Show Commands on page 92

## Create IPv6 Routing Tables

The JUNOS software can maintain one or more routing tables, allowing the software to store route information learned from different protocols separately. For example, it is common for the routing software to maintain unicast routes and multicast routes in different routing tables. You also might have policy considerations that would lead you to create separate routing tables to manage the propagation of routing information.

The JUNOS software uses the default IPv6 routing table, which is inet6.0 for unicast routes.

To configure an IPv6 routing table, include the rib statement at the [edit routing-options] hierarchy level:

```
[edit routing-options]
rib inet6.0 {
  martians {
    destination-prefix match-type <allow>;
  }
  static {
    defaults {
      static-options;
    }
    rib-group group-name;
    route destination-prefix {
      next-hop;
      qualified-next-hop address {
        metric metric;
        preference preference;
      }
      static-options;
    }
  }
}
```

The routing table name, inet6.0, is the primary IPv6 routing table. The protocol family must be inet6 for the IPv6 address family. The number represents the instance of the routing table. For example, the primary instance is 0.

### **Example: Configure an IPv6 Routing Table**

Configure the routing table inet6.0 and add a static route to it:

```
[edit routing-options]
rib inet6.0 {
  static {
    route 8:1::1/128 next-hop 8:3::1;
  }
}
```

## Configure Static Routes

The router uses dynamic routes to learn how to reach network destinations. Dynamic routes are determined from the information exchanged by the routing protocols and, as the name implies, the routes might change as network conditions change and these changes are discovered by the routing protocols. You can configure static (nonchanging) routes to some network destinations. The router uses static routes when it does not have a route to a destination that has a better (lower) preference value, when it cannot determine the route to a destination, or when it is forwarding unroutable packets.

A static route is installed in the routing table only when the route is active. The route is considered active when the list of next-hop routers configured for that route contains at least one next hop on an operational interface, or if you configure it as a discard or reject route.

You can add the same routes to more than one routing table.

To explicitly configure static routes in the primary IPv6 routing table (inet6.0), include the static statement at the [edit routing-options rib inet6.0] hierarchy level:

```
[edit routing-options]
rib inet6.0 {
  static {
    defaults {
      static-options;
    }
    rib-group group-name;
    route destination-prefix {
      next-hop;
      qualified-next-hop address {
        metric metric;
        preference preference;
      }
      static-options;
    }
  }
}
```

The static statement consists of two parts:

**defaults**—Specify global static route options. These are treated as global defaults and apply to all the static routes you configure in the static statement. This part of the static statement is optional.

**route**—Configure individual static routes. In this part of the static statement, you optionally can configure static route options. These options apply to the individual destination only and override any options you configured in the defaults part of the static statement.

The following sections explain how to configure static routes:

Specify the Destination of the Static Route on page 76

Specify the Next Hop of the Static Route on page 76

Specify an Independent Preference for a Static Route on page 77

Specify Static Route Options on page 78

- Configure a Static Route on page 82

- Propagate Static Routes into Routing Protocols on page 83

- Example: Configure Static Routes on page 83

- Example: Resolve Route to a Non-Next-Hop Prefix on page 83

## ***Specify the Destination of the Static Route***

When you configure an individual static route in the route part of the static statement, specify the destination of the route (in route *destination-prefix*) in one of the following ways:

*network/mask-length*, where *network* is the network portion of the 128-bit IPv6 address and *mask-length* is the destination prefix length.

## ***Specify the Next Hop of the Static Route***

When you configure an individual static route in the route part of the static statement, specify how to reach the destination (in *next-hop*) in one of the following ways:

*next-hop address*—Address of the next hop to the destination, specified as follows:

128-bit IPv6 address of the next hop

Interface name (for point-to-point interfaces only)

*address/interface-name* to specify an IP address on an interface

*reject*—Do not forward packets addressed to this destination. Instead, drop the packets, send Internet Control Message Protocol version 6 (ICMPv6) unreachable messages to the packets' originators, and install a reject route for this destination into the routing table.

*discard*—Do not forward packets addressed to this destination. Instead, drop the packets, do not send ICMPv6 unreachable messages to the packets' originators, and install a reject route for this destination into the routing table.

*receive*—Cause packets to the destination to be received by the local router.

## Specify an Independent Preference for a Static Route

Configuring independent preferences allows you to configure multiple static routes to the same destination using different next hops with different preferences and metrics. The static route with the best preference, metric, and reachable next hop is chosen as the active route. This feature allows you to specify preference and metric on a next-hop basis using the qualified-next-hop statement.



**Note**

The preference and metric configured using this statement apply only to the qualified next hops. The qualified next-hop preference and metric values override the route preference and metric (for that specific qualified next hop), similar to how route preference overrides the default preference and metric (for that specific route).

To specify an independent preference for a static route, include the following statements at the [edit routing-options rib inet6.0 static route] hierarchy level:

```
[edit routing-options rib inet6.0 static route]
qualified-next-hop address {
  metric metric;
  preference preference;
}
```

The preference value can range from 1 through 255, with a lower number indicating a more preferred route. The metric value can range from 1 through 65,535.



**Note**

The qualified-next-hop statement is mutually exclusive with all other types of next hops, except for next-hop address. Therefore, you cannot include the next-hop reject, next-hop discard, and next-hop receive statements and the qualified-next-hop statement for the same destination.

### Example: Configure Qualified Next Hops

Configure the following qualified next hops:

A static route to fec0:1:1:4::/64 with a next hop through fec0:1:1:2::1, with a metric 10 and preference 10

A static route to fec0:1:1:5::/64 with a next hop through fec0:1:1:2::2, with a metric 6 and preference 5

A static route to fec0:1:1:5::/64 with a next hop through fec0:1:1:2::3, with a metric 6 and preference 7

```
[edit]
routing-options {
  rib inet6.0 {
    static {
      defaults {
        metric 10;
        preference 10;
      }
      route fec0:1:1:4::/64 {
```

```

    next-hop fec0:1:1:2::1 {
    retain;
    no-readvertise;
    }
    route fec0:1:1:5::/64 {
    next-hop fec0:1:1:2::3;
    qualified-next-hop fec0:1:1:2::2 {
    preference 5;
    }
    metric 6;
    preference 7;
    }
  }
}

```

## Specify Static Route Options

In the defaults and route parts of the static statement, you can specify *static-options*, which define additional information about static routes that is included with the route when it is installed in the routing table. All static options are optional. Static options that you specify in the defaults part of the static statement are treated as global defaults and apply to all the static routes you configure in the static statement. Static options that you specify in the route part of the static statement override any global static options and apply to that destination only.

To configure static route options, include one or more of them in the [edit routing-options rib inet6.0 static (defaults | route)] hierarchy level. Each of these options is explained in the sections that follow.

```

[edit routing-options]
rib inet6.0 {
  static {
    defaults {
      (active | passive);
      as-path <as-path> <origin (egp | igp | incomplete)> <atomic-aggregate>
        <aggregator as-number in-address>;
      community [ community-ids ];
      (install | no-install);
      metric metric <type type>;
      (preference | preference2 | color | color2) preference <type type>;
      (readvertise | no-readvertise);
      (no-resolve | resolve);
      (no-retain | retain);
    }
    rib-group group-name;
    route destination-prefix {
      (active | passive);
      as-path <as-path> <origin (egp | igp | incomplete)> <atomic-aggregate>
        <aggregator as-number in-address>;
      community [ community-ids ];
      (install | no-install);
      metric metric <type type>;
      (preference | preference2 | color | color2) preference <type type>;
      (readvertise | no-readvertise);
      (no-resolve | resolve);
      (no-retain | retain);
    }
  }
}

```

The following sections explain how to specify static route options:

Specify the Route Metric on page 79

Specify the Route Preference on page 79

Specify Community Information on page 79

Specify the AS Path on page 80

Specify Whether the Route Is Installed in the Forwarding Table on page 81

Specify Whether the Route Is Permanently Installed in the Forwarding Table on page 81

Specify Whether Inactive Routes Are Removed from the Routing or Forwarding Table on page 81

Specify When the Route Can Be Readvertised on page 82

Specify When the Route Can Be Resolved to a Prefix Not Directly Connected on page 82

### **Specify the Route Metric**

To associate a metric value with a route, include the metric statement at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
metric metric <type type>;
```

### **Specify the Route Preference**

By default, static routes have a preference value of 5. To modify the default preference value, specify a primary preference value (preference). You also can specify a secondary preference value (preference2) and colors, which are even finer-grained preference values (color and color2). To do this, include one or more of the following statements at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
(preference | preference2 | color | color2) preference <type type>;
```

The preference value can be a number from 1 through 255, with a lower number indicating a more preferred route. For more information about preference values, see the *JUNOS Internet Software Configuration Guide: Routing and Routing Protocols*. Note that if you configure an independent preference value, this value overrides the route preference.

In the type option, you can specify the type of route.

### **Specify Community Information**

By default, no Border Gateway Protocol (BGP) community information is associated with static routes. To associate community information with the routes, include the community statement at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
community [ community-ids ];
```

*community-ids* is one or more community identifiers for either communities or extended communities.

The format for community identifiers is:

```
as-number:community-value
```

*as-number* is the autonomous system (AS) number and can range from 1 through 65,534. *community-value* is the community identifier and can range from 0 through 65,535.

You also can specify *community-ids* as one of the following well-known community names, which are defined in RFC 1997:

*no-advertise*—Routes containing this community name are not advertised to other BGP peers.

*no-export*—Routes containing this community name are not advertised outside a BGP confederation boundary.

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

You can also explicitly exclude BGP community information with a static route using the *none* option. Include *none* when configuring an individual route in the route portion of the static statement to override a community option specified in the defaults portion of the statement.

## Specify the AS Path

By default, no AS path information is associated with static routes. To associate AS path information with the routes, include the *as-path* statement at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
as-path <as-path> <origin (egp | igp | incomplete)> <atomic-aggregate>
  <aggregator as-number in-address>;
```

*as-path* is the AS path to include with the route. It can include a combination of individual AS path numbers and AS sets. Enclose sets in brackets ( [ ] ). The first AS number in the path represents the AS immediately adjacent to the local AS. Each subsequent number represents an AS that is progressively farther from the local AS, heading toward the origin of the path.

You also can specify the AS path using the BGP origin attribute, which indicates the origin of the AS path information:

*igp*—Path information originated within the local AS.

*egp*—Path information originated in another AS.

*incomplete*—Path information learned by some other means.

To attach the BGP ATOMIC\_AGGREGATE path attribute to the static route, specify the *atomic-aggregate* statement. This path attribute indicates that the local system selected a less specific route rather than a more specific route.

To attach the BGP AGGREGATOR path attribute to the static route, specify the aggregator statement. When using this statement, you must specify the last AS number that formed the static route (encoded as two octets), followed by the IPv6 address of the BGP system that formed the static route.

### ***Specify Whether the Route Is Installed in the Forwarding Table***

By default, the JUNOS software installs all active static routes into the forwarding table. To configure the software not to install active static routes into the forwarding table, include the no-install statement at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
no-install;
```

Even if you configure a route so it is not installed in the forwarding table, the route is still eligible to be exported from the routing table to other protocols. To explicitly install the routes into the forwarding table, include the install statement at the [edit routing-options static (defaults | route)] hierarchy level. Include this statement when configuring an individual route in the route portion of the static statement to override a no-install statement specified in the defaults portion of the statement.

```
[edit routing-options rib inet6.0 static (defaults | route)]
install;
```

### ***Specify Whether the Route Is Permanently Installed in the Forwarding Table***

By default, statically configured routes are deleted from the forwarding table when the routing protocol process shuts down normally. To have a static route remain in the forwarding table, include the retain statement at the [edit routing-options static (defaults | route)] hierarchy level. Doing this greatly reduces the time required to restart a system that has a large number of routes in its routing table.

```
[edit routing-options rib inet6.0 static (defaults | route)]
retain;
```

To explicitly specify that routes are deleted from the forwarding table, include the no-retain statement at the [edit routing-options static (defaults | route)] hierarchy level. Include this statement when configuring an individual route in the route portion of the static statement to override a retain statement specified in the defaults portion of the statement.

```
[edit routing-options rib inet6.0 static (defaults | route)]
no-retain;
```

### ***Specify Whether Inactive Routes Are Removed from the Routing or Forwarding Table***

By default, static routes are removed from the routing and forwarding tables when they become inactive. To have a static route remain continually installed in the routing and forwarding tables, include the passive statement at the [edit routing-options static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
passive;
```

Routes that have been configured to remain continually installed in the routing and forwarding tables are marked with reject next hops when they are inactive.

To explicitly remove static routes when they become inactive, include the `active` statement at the `[edit routing-options static (defaults | route)]` hierarchy level. Include this statement when configuring an individual route in the `route` portion of the static statement to override a `retain` statement specified in the `defaults` portion of the statement.

```
[edit routing-options rib inet6.0 static (defaults | route)]
  active;
```

### **Specify When the Route Can Be Readvertised**

By default, static routes are eligible to be readvertised (that is, exported) by dynamic routing protocols. To mark a static route as being ineligible for readvertisement, include the `no-readvertise` statement at the `[edit routing-options static (defaults | route)]` hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
  no-readvertise;
```

To explicitly readvertise static routes, include the `readvertise` statement at the `[edit routing-options static (defaults | route)]` hierarchy level. Include the `readvertise` statement when configuring an individual route in the `route` portion of the static statement to override a `retain` option specified in the `defaults` portion of the statement.

```
[edit routing-options rib inet6.0 static (defaults | route)]
  readvertise;
```

### **Specify When the Route Can Be Resolved to a Prefix Not Directly Connected**

By default, static routes can point only to a directly connected next hop. You can configure a route to a prefix that is not directly connected by resolving the route through the `inet6.0` routing table. To configure a static route to a prefix that is not a directly connected next hop, include the `resolve` statement at the `[edit routing-options rib inet6.0 static (defaults | route)]` hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
  resolve;
```

To not allow a static route to resolve to a non-next-hop prefix, include the `no-resolve` statement at the `[edit routing-options rib inet6.0 static (defaults | route)]` hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
  no-resolve;
```

### **Configure a Static Route**

To configure a static route, include the next-hop address and `retain` statements at the `[edit routing-options rib inet6.0 static (defaults | route)]` hierarchy level:

```
[edit routing-options rib inet6.0 static (default | route)]
  next-hop address;
  retain;
```

## Propagate Static Routes into Routing Protocols

A common way to propagate static routes into the various routing protocols is to configure the routes so that the next-hop router is the loopback address. However, configuring static routes in this way with the JUNOS software does not propagate the static routes, because the forwarding table ignores static routes whose next-hop router is the loopback address. To propagate static routes into the routing protocols, include the discard statement at the [edit routing-options rib inet6.0 static (defaults | route)] hierarchy level:

```
[edit routing-options rib inet6.0 static (defaults | route)]
discard;
```

In this configuration, you use the discard statement instead of reject because discard does not send an ICMPv6 unreachable message for each packet that it drops.

## Example: Configure Static Routes

Configure a default route through the next-hop router 8:3::1:

```
[edit]
user@host# set routing-options rib inet6.0 static route abcd::/48 next-hop 8:3::1
[edit]
user@host# show
routing-options {
  static {
    route abcd::/48 next-hop 8:3::1;
  }
}
```

## Example: Resolve Route to a Non-Next-Hop Prefix

Resolve a static route to non-next-hop 1::/64 using next-hop 2000::1:

```
[edit]
user@host# set routing-options rib inet6.0 static route 1::/64 next-hop 2000::1 resolve
[edit]
user@host# show route 1::/64
inet6.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

1::/64          *[Static/5] 00:01:50
                > to 8:1::2 via ge-0/1/0.0

user@host# show route 2000::1
inet6.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

2000::/126     *[BGP/170] 00:05:32, MED 20, localpref 100
                AS path: 2 I
                > to 8:1::2 via ge-0/1/0.0
```

## Configure Martian Addresses

Martian addresses are host or network addresses about which all routing information is ignored. They commonly are sent by improperly configured systems on the network and have destination addresses that are obviously invalid.

In IPv6, the following are the default martian addresses: loopback address, the reserved and unassigned prefixes from RFC 2373, and the link-local unicast prefix.

### Add Martian Addresses

To add martian addresses to the list of default martian addresses in any other routing tables, or to explicitly add martian addresses to the list of default martian addresses in the primary IPv6 routing table (inet6.0), include the martians statement at the [edit routing-options rib inet6.0] hierarchy level:

```
[edit]
routing-options {
  rib inet6.0 {
    martians {
      destination-prefix match-type;
    }
  }
}
```

In *destination-prefix*, specify the routing destination in one of the following ways:

*network/mask-length*—*network* is the network portion of the IPv6 address and *mask-length* is the destination prefix length.

In *match-type*, specify the type of match to apply to the destination prefix.

### Remove Martian Addresses

To delete a martian address from within a range of martian addresses, include the *allow* statement in the martians statement. This statement removes an exact prefix that is within a range of addresses specified to be martian addresses.

To delete a martian address from any other routing tables, or to explicitly delete a martian address from the primary IPv6 routing table (inet6.0), include the martians statement at the [edit routing-options rib inet6.0] hierarchy level:

```
[edit]
routing-options {
  rib inet6.0 {
    martians {
      destination-prefix match-type allow;
    }
  }
}
```

## Create Routing Table Groups

You can group together one or more routing tables to form a *routing table group*. Within a group, a routing protocol can import routes into all the routing tables in the group and can export routes from a single routing table.

To create a routing table group, include the `rib-groups` statement at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  rib-groups group-name {
    import-policy [ policy-names ];
    import-rib [ routing-table-names ];
    export-rib routing-table-name;
  }
}
```

The routing table group can have any name you choose (specified in *group-name*). If the group name you specify is not created explicitly, naming it in the `rib-groups` statement creates it.

Each routing table group must contain one or more routing tables that the JUNOS software uses when importing routes (specified in the `import-rib` statement). The first routing table you specify is the *primary routing table*, and any additional routing tables are *secondary routing tables*. The primary routing table determines the address family of the routing table group. To configure an IPv6 routing table group, specify `inet6.0` as the primary routing table. If you configure an IPv6 routing table group, the primary and all secondary routing tables must be IPv6 routing tables (`inet6.x`). You cannot have `inet` and `inet6` routing tables in the same `import-rib` statement.

Each routing table group optionally can contain one routing table group that the JUNOS software uses when exporting routes to the routing protocols (specified in the `export-rib` statement).

After specifying the routing table from which to import routes, you can apply one or more policies to control which routes will be installed in the routing table group. To apply a policy to routes being imported into the routing table group, include the `import-policy` statement at the [edit routing-options rib-groups *group-name*] hierarchy level:

```
[edit]
routing-options {
  rib-groups group-name {
    import-policy [ policy-names ];
  }
}
```

### Example: Create a Routing Table Group

Create a routing table group so that interface routes are installed into two routing tables, inet6.0 and inet6.2:

```
[edit]
routing-options {
  interface-routes {
    rib-group inet6 if-rg;
  }
  rib-groups if-rg {
    rib-groups if-rg {
      import-rib [ inet6.0 inet6.2 ];
    }
  }
}
```

### Configure How Interface Routes Are Imported into Routing Tables

By default, interface routes (also called direct routes) are imported into the primary IPv6 routing table inet6.0. To import interface routes into a different routing table, you create an IPv6 routing table group and associate it with the router's interfaces.

To associate an IPv6 routing table group to an interface, include the interface-routes statement at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  interface-routes {
    rib-group inet6 group-name;
  }
}
```

### Configure a Route Limit for Routing Tables

A route limit sets an upper limit for the number of prefixes installed in routing tables. A route limit applies only to dynamic routing protocols, not to static or interface routes.

To configure a route limit, include the maximum-routes statement at the [edit routing-options] hierarchy level:

```
[edit routing-options]
maximum-routes route-limit <log-only | threshold value>;
```

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

## Install a Static Route into More than One Routing Table

You can install a static route into more than one routing table. For example, you might want a simple configuration that allows you to install a static route into the primary IPv6 routing table inet6.0, as well as a second routing table inet6.2. Instead of configuring the same static route for each routing table, you can insert the route into multiple tables using routing table groups. To create a routing table group, include the rib-group statement at the [edit routing-options rib *routing-table-name* static] hierarchy level:

```
[edit routing-options rib routing-table-name static]
  rib-group group-name;
```

To install the routing table into a configured routing table group, include the import-rib statement at the [edit routing-options rib-groups] hierarchy level:

```
[edit routing-options rib-groups]
  import-rib [ routing-table-names ]
```

The first routing table you list in the import-rib statement must be the one you configured in the rib-group statement.

### **Example: Install a Static Route into More than One Routing Table**

Install a static route into the inet6.0 and inet6.2 routing tables:

```
[edit routing-options rib table1.inet6.0 static]
  rib-group groupA;

[edit routing-options rib-groups]
  groupA {
    import-rib [table1.inet6.0 inet6.0 inet6.2]
  }
```

## Configure the AS Number

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

ASs are identified by a number from 1 through 65,535 that is assigned by the Network Information Center (NIC) (in the United States, <http://www.isi.edu>).

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

To configure the router's AS number, include the autonomous-system statement at the [edit routing-options] hierarchy level:

```
[edit]
  routing-options {
    autonomous-system autonomous-system <loops number>;
  }
```

To specify how many times this AS number can appear in an AS path, include the loops statement.

## Configure the Router Identifier

The router identifier is used by BGP to identify the router from which a packet originated. The router identifier usually is the IPv4 address of the local router. If you do not configure a router identifier, the IPv4 address of the first interface encountered in the router is used.

To configure the router identifier, include the `router-id` statement at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  router-id address;
}
```

## Configure Graceful Restart

Graceful restart allows a router undergoing a restart to inform its adjacent neighbors and peers of its condition. Graceful restart is disabled by default.

To enable graceful restart, include the `graceful-restart` statement at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  graceful-restart {
    disable;
    path-selection-defer-time-limit time-limit;
  }
}
```

## Configure AS Confederation Members

If you administer multiple ASs that contain a very large number of BGP systems, you can group them into one or more *confederations*. Each confederation is identified by its own AS number, which is called a *confederation AS number*. To external ASs, a confederation appears to be a single AS. Thus, the internal topology of the ASs making up the confederation is hidden.

The BGP path attributes NEXT\_HOP, LOCAL\_PREF, and MULTI\_EXIT\_DISC, which normally are restricted to a single AS, are allowed to be propagated throughout the ASs that are members of the same confederation.

Because each confederation is treated as if it were a single AS, you can apply the same routing policy to all the ASs that make up the confederation.

Grouping ASs into confederations reduces the number of BGP connections required to interconnect ASs.

If you are using BGP, you can enable the local router to participate as a member of an AS confederation. To do this, include the `confederation` statement at the [edit routing-options] hierarchy level:

```
[edit]
routing-options {
  confederation confederation-autonomous-system members [ autonomous-systems ];
}
```

Specify the AS confederation identifier, along with the AS numbers that are members of the confederation.

Note that peer adjacencies will not form if two BGP neighbors disagree about whether an adjacency falls within a particular confederation.

## Configure Logging for the Routing Protocol Process

To control how much information the routing protocol process should log, include the options statement at the [edit routing-options] hierarchy level. Include the following form of the statement to log messages at one or more individual severity levels:

```
[edit]
routing-options {
  options syslog level;
}
```

Include the following form of the statement to log messages for a particular severity level and higher (more severe) levels, where level 7 (debug) is least severe and level 0 (emergency) is most severe:

```
[edit]
routing-options {
  options syslog upto level;
}
```

### **Examples: Configure System Logging for the Routing Protocol Process**

Configure the router to log messages of all severities:

```
[edit]
user@host# set routing-options options syslog upto emergency
[edit]
user@host# show
routing-options {
  options syslog upto emergency;
}
```

Configure the router to log only alert-level and critical-level messages:

```
[edit]
user@host# set routing-options options syslog alert critical
[edit]
user@host# show
routing-options {
  options syslog alert critical;
}
```

## Trace Global Routing Protocol Operations

Global routing protocol tracing operations track all general routing operations and record them in a log file. Any global tracing operations that you configure are inherited by the individual routing protocols. To modify the global tracing operations for an individual protocol, configure tracing when configuring that protocol.

For a general discussion of tracing and of the precedence of multiple tracing operations, see tracing and logging operations in the *JUNOS Internet Software Configuration Guide: Getting Started*.

To trace the paths of multicast packets, use the `mtrace` command, as described in the *JUNOS Internet Software Operational Mode Command Reference*.

To configure global routing protocol tracing flags, include the `traceoptions` statement at the [edit routing-options] hierarchy level:

```
[edit routing-options]
traceoptions {
  file name <replace> <size size> <files number> <no-stamp>
    <(world-readable | no-world-readable)>;
  flag flag <flag-modifier> <disable>;
}
```

You can specify the following global routing protocol tracing flags:

`all`—Trace all tracing operations.

`general`—Trace all normal operations and routing table changes (a combination of the normal and route trace operations).

`normal`—Trace all normal operations.

`policy`—Trace policy operations and actions.

`route`—Trace routing table changes.

`state`—Trace state transitions.

`task`—Trace interface transactions and processing.

`timer`—Trace timer usage.

You can specify the following tracing flag modifiers:

`detail`—Provide detailed trace information.

`receive`—Trace only packets being received.

`send`—Trace only packets being transmitted.

The flags in a traceoptions flag statement are identifiers. When you use the set command to configure a flag, any flags that might already be set are not modified. In the following example, setting the csN tracing flag has no effect on the already configured detail flag. Use the delete command to delete a particular flag.

```
[edit protocols isis]
user@host# show
traceoptions {
  flag csN detail;
}
[edit protocols isis]
user@host# set traceoptions flag csN
[edit protocols isis]
user@host# show
traceoptions {
  flag csN detail;
}
user@host# delete traceoptions flag detail
[edit protocols isis]
user@host# show
traceoptions {
  flag csN;
}
```

### **Examples: Configure Global Tracing Operations**

Log all globally traceable operations, saving the output in up to 10 files that are up to 10 MB in size:

```
[edit]
routing-options {
  traceoptions {
    file routing size 10m files 10;
    flag all;
  }
}
```

Log all unusual or abnormal traceable operations:

```
[edit]
routing-options {
  traceoptions {
    file routing size 10m files 10;
    flag all;
    flag normal disable;
  }
}
```

Log changes that occur in the JUNOS software routing table:

```
[edit]
routing-options {
  traceoptions {
    file routing size 10m files 10;
    flag route;
  }
}
```

## Route Show Commands

From the command-line interface (CLI), you can run show commands to display routing table information.

### ***show route forwarding-table***

**Syntax** show route forwarding-table <destination-prefix> <family family | matching matching> <multicast> <vpn vpn> <detail | summary>

**Description** Display the route entries in the kernel's forwarding table. This is the version of the forwarding table in the Routing Engine. The Routing Engine copies this table to the Packet Forwarding Engine.

The show **route forwarding-table** command displays only the network-layer prefixes and their next hops. Usually, you need information in the show route commands. However, when you run into problems, such as connectivity problems, you can use the show **route forwarding-table** command to verify that the routing protocol process has relayed the correct information to the forwarding table.



**Note**

The show route forwarding-table command is an independent command, not a filter that selects specific information that is displayed from the routing tables. You cannot use this command in conjunction with any of the show route filter options.

**Options** none—Display the routes in the forwarding table. This option is equivalent to the UNIX netstat -rn command.

*destination-prefix*—(Optional) IP address of a destination.

*detail*—(Optional) Display detailed information, including clone routes. This option is equivalent to the UNIX netstat -rna command.

*family family*—(Optional) Display routing table entries for the specified family.

*matching matching*—(Optional) Display routing table entries matching the specified prefix or prefix length.

*multicast*—(Optional) Display routing table entries for multicast routes.

*summary*—(Optional) Display summary information about routes.

**Required Privilege Level** view

|                      |  |
|----------------------|--|
| <b>Output Fields</b> | Internet, ISO, MPLS—IP, Internet Organization for Standardization (ISO), and Multiprotocol Label Switching (MPLS) addresses. |
|                      | Destination—Destination of the route.  |
|                      | Type—How the route was placed into the forwarding table (located in the second column):                                      |
|                      | clon—Clone route (for TCP or multicast only; displayed only if you specify the detail option).                               |
|                      | dest—Remote addresses directly reachable through an interface.   |
|                      | iddn—dest route for which the interface is down.   |
|                      | ifcl—clon route for which the interface is down.   |
|                      | ifdn—intf route for which the interface is down.   |
|                      | ignr—Ignore this route.  |
|                      | intf—Installed as the result of configuring an interface.  |
|                      | perm—Permanent (installed by the kernel when the routing table is initialized).  |
|                      | user—Installed by the routing protocol process or as a result of the configuration.  |
|                      | RtRef—IP, ISO, and MPLS addresses.   |
|                      | InIf—Incoming interface.   |
|                      | Flags—Additional information about a route.  |
|                      | 0x01—Static route.   |
|                      | 0x02—Cache route.  |
|                      | 0x04—Check against incoming interface.   |
|                      | 0x08—Route has accounting.   |
|                      | 0x10—Route has been sent to the Packet Forwarding Engine.  |
|                      | NextHop—Next hop to the destination.   |
|                      | Type—Information about the next hop (located in the fourth column):  |
|                      | bcst—Broadcast.  |
|                      | dscd—Discard; no ICMP unreachable message sent.  |
|                      | hold—Next hop is waiting to be resolved (will turn into a unicast or multicast).   |
|                      | locl—Local address on an interface.  |
|                      | mcrt—“Regular” multicast next hop.   |
|                      | mcst—Wire multicast next hop (limited to the LAN).   |

mdsc—Multicast discard.

mgrp—Multicast group member.

recv—Receive.

rjct—Discard; ICMP unreachable message sent.

rslv—Resolving next hop.

ucst—Unicast.

ulst—List of unicast next hops. A packet sent to this next hop will go to any next hop in the list.

Index—Number associated with the next hop.

NhRef—Number of routes that reference this next hop.

Netif—Interface used to reach the next hop.

indr—An index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.

#### Sample Output

```
user@host> show route forwarding-table
Internet:
Destination          Type RtRef Nexthop          Type Index NhRef Netif
default              user  1 111.222.1.254          ucst  20   3 de0.0
default              perm  0                      rjct   9    1
10.168.1.8           intf  0 10.168.1.8            locl  16    1
111.222/23           intf  0                      rslv  14    1 de0.0
111.222.0.0          dest  0 111.222.0.0           recv  12    1 de0.0
111.222.1.8          intf  0 111.222.1.8           locl  13    2
111.222.1.8          dest  0 111.222.1.8           locl  13    2
111.222.1.254        dest  0 0:a0:c9:69:ea:f3      ucst  20    3 de0.0
111.222.1.255        dest  0 111.222.1.255         bcst  11    1 de0.0
111.222.7.4/30       intf  0 ff.3.0.21             ucst  22    1 mps0.0
111.222.7.5          intf  0 111.222.7.5           locl  21    1
224/4                perm  0                      mdsc   8    1
224.0.0.1            perm  0 224.0.0.1             mcst   4    1
255.255.255.255     perm  0                      bcst   5    1

ISO:
Destination          Type RtRef Nexthop          Type Index NhRef Netif
default              perm  0                      dscd  15    1
47.0005.80ff.f800.0000.0108.0001.1921.6800.1008.00
                    intf  0                      locl  17    1
```

#### Sample Output

```
user@host> show route forwarding-table detail
Internet:
Destination          Type RtRef InIf  Flags Nexthop          Type Index NhRef Net
if
default              user  0    0  0x10                      rjct   9    9
default              perm  0    0  0x00                      rjct   9    9
1.1.1.8/32           user  0    0  0x10 111.222.8.208       ucst  35    5 so-
2/1/1.0
1.1.1.9/32           user  0    0  0x10 111.222.8.208       ucst  35    5 so-
2/1/1.0
10.0.222.0/30        ifdn  0    0  0x00 ff.3.0.21           ucst  51    1 t3-
5/2/2.0
```

```

10.0.222.1/32      user  0  0  0x10          rjct  9  9  .
10.0.222.1/32      intf  0  0  0x00 10.0.222.1  locl  50  1  .
10.1.1.76/32       user  0  0  0x10 111.222.8.208 ucst  35  5 so- .
2/1/1.0
10.255.245.87/32  user  0  0  0x10 111.222.8.208 ucst  35  5 so- .
2/1/1.0
12.12.12.0/24      ifdn  0  0  0x00 12.12.12.0   rjct  29  1 so- .
2/1/0.1
12.12.12.0/32      iddn  0  0  0x00 12.12.12.0   recv  31  1 so- .
2/1/0.1
12.12.12.12/32     user  0  0  0x10          rjct  9  9  .
12.12.12.12/32     intf  0  0  0x00 12.12.12.12  locl  28  1  .
2/1/0.1
12.12.12.14/32     iddn  0  0  0x00 dlci: 14   ucst  33  1 so- .
2/1/0.1
12.12.12.255/32    iddn  0  0  0x00 12.12.12.255  recv  30  1 so- .
2/1/0.1
127.0.0.1/32       intf  0  0  0x10 127.0.0.1     locl  16  1  .
111.222.0.0/16     user  0  0  0x10 111.222.4.254 ucst  19  6 fxp .
0.0
111.222.4.0/24     intf  0  0  0x10          rslv  14  1 fxp .
0.0
111.222.4.0/32     dest  0  0  0x00 111.222.4.0   recv  12  1 fxp .
0.0
111.222.4.21/32    intf  0  0  0x10 111.222.4.21  locl  13  2  .
111.222.4.21/32    dest  0  0  0x00 111.222.4.21  locl  13  2  .
111.222.4.254/32   dest  0  0  0x00 0:0:c:6:4b:a1  ucst  19  6 fxp .
0.0
111.222.4.255/32   dest  0  0  0x00 111.222.4.255  bcst  11  1 fxp .
0.0
111.222.8.16/28    intf  0  0  0x10 111.222.8.16   rjct  41  1 at- .
2/3/0.0
111.222.8.16/32    dest  0  0  0x10 111.222.8.16   recv  43  1 at- .
2/3/0.0
111.222.8.18/32    dest  0  0  0x10 0.0.20.88.aa.aa.3.0.0.0.8.0 ucst  44  .
1 at-2/3/0.0
111.222.8.19/32    dest  0  0  0x10 0.1.20.a8.aa.aa.3.0.0.0.8.0 ucst  45  .
1 at-2/3/0.0
111.222.8.31/32    dest  0  0  0x10 111.222.8.31   recv  42  1 at- .
2/3/0.0
111.222.8.96/30    intf  0  0  0x10 0.3.10.c8.aa.aa.3.0.0.0.8.0 ucst  49  .
1 at-2/3/0.130
111.222.8.97/32    intf  0  0  0x10 111.222.8.97   locl  48  1  .
111.222.8.104/30   intf  0  0  0x10 0.2.10.88.aa.aa.3.0.0.0.8.0 ucst  47  .
1 at-2/3/0.128
111.222.8.106/32   intf  0  0  0x10 111.222.8.106  locl  46  1  .
111.222.8.112/30   intf  0  0  0x10 ff.3.0.21     ucst  21  1 so- .
2/0/1.0
111.222.8.114/32   intf  0  0  0x10 111.222.8.114  locl  20  1  .
111.222.8.116/30   intf  0  0  0x10 ff.3.0.21     ucst  23  1 so- .
2/0/2.0
111.222.8.118/32   user  0  0  0x10          rjct  9  9  .
111.222.8.118/32   intf  0  0  0x00 111.222.8.118  locl  22  1  .
111.222.8.120/30   intf  0  0  0x10 ff.3.0.21     ucst  25  1 so- .
2/0/3.0
111.222.8.122/32   user  0  0  0x10          rjct  9  9  .
111.222.8.122/32   intf  0  0  0x00 111.222.8.122  locl  24  1  .
111.222.8.204/32   user  0  0  0x10          rjct  9  9  .
111.222.8.204/32   intf  0  0  0x00 111.222.8.204  locl  38  1  .
111.222.8.205/32   ifdn  0  0  0x00 ff.3.0.21     ucst  39  1 so- .
2/1/3.0

```



**show route receive-protocol**

**Syntax** show route receive-protocol *protocol neighbor-address* < detail | extensive | standard >

**Description** Display the routing information as it was received through a particular neighbor of a particular dynamic routing protocol. This information includes the routes that the local router advertised to the neighbor. The information displayed reflects the routes before they are filtered by that protocol's import routing policy statements and is placed into the routing table.

**Options** none—Display information that was received from the neighbor.

detail—(Optional) Display additional information that was received from the neighbor.

extensive—(Optional) Display extensive information that was received from the neighbor.

*neighbor-address*—Address of the neighboring router from which the route entry was received.

*protocol*—Protocol transmitting the route. It can be one of the following: bgp or ripng.

**Required Privilege Level** view

**Output Fields** *routing-table-name*—Name of the routing table; for example, inet6.0.

destinations—Number of destinations for which there are routes in the routing table.

routes—Number of routes in the routing table:

active—Number of routes that are active.

holddown—Number of routes that are in the hold-down state prior to being declared inactive.

hidden—Number of routes not used because of routing policy.

Prefix—Route address.

Next hop—Address of the next hop to the address.

MED—Multiple exit discriminator (MED) value included in the route.

LcIpref—Local preference value included in the route.

AS path—AS path included in the route.

Route Distinguisher—IP subnets are augmented with a 64-bit prefix called a route distinguisher to make them unique.

VPN Label—Label assigned to this Virtual Private Network (VPN).

Communities—Community path attribute of the route.

AS path: I <Originator>—(For route reflected output only) Route reflector set the originator ID attribute.

Cluster list—(For router reflected output only) Cluster IDs sent by the route reflector.

Originator ID—(For router reflected output only) Address of router that originally sent route to the route reflector.

```

Sample Output user@host> show route receive-protocol bgp 10.255.245.63 extensive

inet.0: 244 destinations, 244 routes (243 active, 0 holddown, 1 hidden)
Prefix          Nexthop          MED    Lclpref AS path
1.1.1.0/24 (1 entry, 1 announced)
  Nexthop: 10.0.50.3
  Localpref: 100
  AS path: I <Originator>
  Cluster list: 10.2.3.1
  Originator ID: 10.255.245.45
165.3.0.0/16 (1 entry, 1 announced)
  Nexthop: 111.222.5.254
  Localpref: 100
  AS path: I <Originator>
  Cluster list: 10.2.3.1
  Originator ID: 10.255.245.68
165.4.0.0/16 (1 entry, 1 announced)
  Nexthop: 111.222.5.254
  Localpref: 100
  AS path: I <Originator>
  Cluster list: 10.2.3.1
  Originator ID: 10.255.245.45
195.1.1.2.0/24 (1 entry, 1 announced)
  Nexthop: 111.222.5.254
  Localpref: 100
  AS path: I <Originator>
  Cluster list: 10.2.3.1
  Originator ID: 10.255.245.68

inet.2: 63 destinations, 63 routes (63 active, 0 holddown, 0 hidden)
Prefix          Nexthop          MED    Lclpref AS path

inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix          Nexthop          MED    Lclpref AS path

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix          Nexthop          MED    Lclpref AS path

mpls.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)

```

**show route table inet6.0**

**Syntax** show route table inet6.0

**Description** Display information about the primary IPv6 routing table.

**Required Privilege Level** view

**Sample Output**

```
user@host> show route table inet6.0
inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Route, * = Both

fec0:0:0:3::/64 *[Direct/0] 00:01:34
                 >via fe-0/1/0.0
fec0:0:0:3::/128 *[Local/0] 00:01:34
                 >Local
fec0:0:0:4::/64 *[Static/5] 00:01:34
                 >to fec0:0:0:3::ffff via fe-0/1/0.0
```

