

Chapter 4

Configure Routing Tables and Routes

This chapter discusses how to perform the following tasks for configuring routing tables and routes:

Creating Routing Tables on page 43

Configuring Static Routes on page 45

Configuring Aggregate Routes on page 66

Configuring Generated Routes on page 74

Configuring Martian Addresses on page 82

Applying a Filter to a Forwarding Table on page 84

Creating Routing Tables

The JUNOS software can maintain one or more routing tables, thus 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.

Creating routing tables is optional. If you do not create any, the JUNOS software uses its default routing tables, which are `inet.0` for Internet Protocol version 4 (IPv4) unicast routes, `inet6.0` for Internet Protocol version 6 (IPv6) unicast routes, `inet.1` for the IPv4 multicast forwarding cache, and `inet.3` for IPv4 Multiprotocol Label Switching (MPLS). If the Multiprotocol Border Gateway Protocol (MBGP) is enabled, `inet.2` is used for Subsequent Address Family Indicator (SAFI) 2 routes. If you configure a routing instance, the JUNOS software creates the default unicast routing table `instance-name.inet.0`.

If you want to add static, aggregate, generated, or martian routes only to the default IPv4 unicast routing table (`inet.0`), you do not have to create any routing tables because, by default, these routes are added to `inet.0`. You can add these routes just by including the `static`, `aggregate`, `generate`, and `martians` statements. For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement.

To explicitly create a routing table, include the rib statement:

```
[edit]
routing-options {
  rib routing-table-name {
    static {
      defaults {
        static-options;
      }
      rib-group group-name;
      route destination-prefix {
        lsp-next-hop lsp-name {
          metric metric;
          preference preference;
        }
        next-hop;
        p2mp-lsp-next-hop {
          metric metric;
          preference preference;
        }
        qualified-next-hop address {
          metric metric;
          preference preference;
        }
      }
      static-options;
    }
  }
  aggregate {
    defaults {
      aggregate-options;
    }
    route destination-prefix {
      policy policy-name;
      aggregate-options;
    }
  }
  generate {
    defaults {
      generate-options;
    }
    route destination-prefix {
      policy policy-name;
      generate-options;
    }
  }
  martians {
    destination-prefix match-type <allow>;
  }
}
}
```

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

The routing table name, *routing-table-name*, includes the protocol family, optionally followed by a period and a number. The protocol family can be `inet` for the IPv4 family, `inet6` for the IPv6 family, or `iso` for the International Standards Organization (ISO) protocol family. The number represents the routing instance. The first instance is 0.

Example: Creating Routing Tables

Create the IPv4 routing table `inet.4` and add a static route to it:

```
[edit]
routing-options {
  rib inet.4 {
    static {
      route 140.122.0.0/16 next-hop 192.168.0.10;
    }
  }
}
```

Configure the primary IPv6 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;
  }
}
```

Configuring 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; that is, the list of next-hop routers configured for that route contains at least one next hop on an operational interface.

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

To configure static routes in the default IPv4 routing table (inet.0), include the static statement:

```
[edit]
routing-options {
  static {
    defaults {
      static-options;
    }
    rib-group group-name;
    route destination-prefix {
      lsp-next-hop lsp-name {
        metric metric;
        preference preference;
      }
      next-hop;
      p2mp-lsp-next-hop {
        metric metric;
        preference preference;
      }
      qualified-next-hop address {
        metric metric;
        preference preference;
      }
      static-options;
    }
  }
}
```

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

To configure static routes in one of the other routing tables, to explicitly configure static routes in the default IPv4 route table (inet.0), or to explicitly configure static routes in the primary IPv6 routing table (inet6.0), include the static statement:

```
[edit]
routing-options {
  rib routing-table-name {
    static {
      defaults {
        static-options;
      }
      rib-group group-name;
      route destination-prefix {
        lsp-next-hop lsp-name {
          metric metric;
          preference preference;
        }
        next-hop;
        p2mp-lsp-next-hop {
          metric metric;
          preference preference;
        }
      }
    }
  }
}
```


Specifying 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 IP address and *mask-length* is the destination prefix length.

default if this is the default route to the destination. This is equivalent to specifying an IP address of 0.0.0.0/0.

Specifying 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*—IPv4 or IPv6 address of the next hop to the destination, specified as:

IPv4 or IPv6 address of the next hop

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

address or *interface-name* to specify an IP address of a multipoint interface or an interface name of a point-to-point interface.



NOTE: If an interface becomes unavailable, all configured static routes on that interface are withdrawn from the routing table.

next-table *routing-table-name*—Name of the next routing table to the destination.

reject—Do not forward packets addressed to this destination. Instead, drop the packets, send ICMP (or 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 ICMP (or 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.

Specifying an Independent Preference for a Static Route

Configuring independent preferences allows you to configure multiple static routes with different preferences and metrics to the same destination. 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 by means of this statement only apply to the qualified next hops. The qualified-next-hop preference and metric override the route preference and metric (for that specific qualified next hop), similar to how the 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:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The preference value can be a number in the range from 1 through 255, with a lower number indicating a more preferred route. The metric value can be a number in the 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 configure next-hop reject, next-hop discard, and next-hop receive with qualified-next-hop for the same destination.

Example: Configuring Independent Preferences for an IPv4 Static Route

The following example configures:

A static route to 0.0.0.0/8 with a next hop through 192.168.1.254, with a metric 10 and preference 10.

A static route to 10.0.0.0/8 with a next hop through 192.168.1.254, with a metric 6 and preference 5.

A static route to 10.0.0.0/8 with a next hop through 192.168.1.2, with a metric 6 and preference 7.

```
[edit]
routing-options {
  static {
    defaults {
      metric 10;
      preference 10;
    }
    route 0.0.0.0/8 {
      next-hop 192.168.1.254 {
        retain;
        no-readvertise;
      }
    }
    route 10.0.0.0/8 {
      next-hop [192.168.1.2];
      qualified-next-hop 192.168.1.254 {
        preference 5;
      }
      metric 6;
      preference 7;
    }
  }
}
```

Example: Configuring Independent Preferences for an IPv6 Static Route

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;
}
}
}
}
}

```

Specifying an LSP as the Next Hop for a Static Route

Static routes can be configured with a next hop that is a label-switched path (LSP). This is useful when implementing filter-based forwarding. You can specify an LSP as the next hop and assign an independent preference and metric to this next hop.

To specify an LSP as the next hop for a static route, include the following statements:

```

[edit routing-options static route]
lsp-next-hop lsp-name {
  metric metric;
  preference preference;
}

```



NOTE: The preference and metric configured by means of the `lsp-next-hop` statement only apply to the LSP next hops. The LSP next-hop preference and metric override the route preference and metric (for that specific LSP next hop), similar to how the route preference overrides the default preference and metric (for that specific route).

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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



NOTE: The `lsp-next-hop` statement is mutually exclusive with all other types of next hops, except for next-hop address and qualified-next-hop. Therefore, you cannot configure next-hop reject, next-hop discard, next-hop receive, and next-table with `lsp-next-hop` for the same destination.

To specify a point-to-multipoint LSP as the next hop for a static route, include the following statements:

```
[edit routing-options static route]
p2mp-lsp-next-hop {
  metric metric;
  preference preference;
}
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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

Installing 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 default routing table inet.0, as well as a second routing table inet.2. Instead of configuring the same static route for each routing table, you can use routing table groups to insert the route into multiple tables. To create a routing table group, include the rib-group statement.

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

To install the routing table into a configured routing table group, include the import-rib statement:

```
rib-group group-name {
  import-rib [ routing-table-names ];
}
```

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

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

Examples: Installing a Static Route into More than One Routing Table

Install an IPv4 static route into inet.0 and inet.2:

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

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

Install an IPv6 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];
}
```

Specifying 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 for IPv4 static routes, include one or more options in the defaults or route part of the static statement. Each of these options is explained in the sections that follow.

```
[edit]
routing-options {
  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);
      (retain | no-retain);
      tag string;
    }
    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);
      resolve;
      (retain | no-retain);
      tag string;
    }
  }
}
```

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

To configure static route options for IPv6 static routes, include one or more options in the defaults or route part of the static statement. 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);
      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);
      resolve;
      (retain | no-retain);
    }
  }
}
```

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

The following sections explain how to specify static route options:

Specifying the Route Metric on page 55

Specifying the Route Preference on page 55

Specifying Community Information on page 56

Specifying the AS Path on page 57

Specifying the OSPF Tag on page 58

Specifying Whether a Route Is Installed in the Forwarding Table on page 58

Specifying Whether the Route Is Permanently Installed in the Forwarding Table on page 58

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table on page 60

Specifying When the Route Can Be Readvertised on page 60

Specifying When the Route Can Be Resolved to a Prefix That Is Not Directly Connected on page 61

Configuring Bidirectional Forwarding Detection on page 61

Specifying the Route Metric

To associate a metric value with an IPv4 route, include the metric statement:

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

To associate a metric value with an IPv6 route, include the metric statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

In the type option, you can specify the type of route. For OSPF, when routes are exported to OSPF, type 1 routes are advertised in type 1 externals, and routes of any other type are advertised in type 2 externals. Note that if a qualified-next-hop metric value is configured, this value will override the route metric.

Specifying 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 for IPv4 static routes, include one or more of the following statements:

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

To do this for IPv6 static routes, include one or more of the following statements:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The preference value can be a number in the range from 1 through 255, with a lower number indicating a more preferred route. For more information about preference values, see “Route Preferences” on page 6. Note that if a qualified-next-hop preference value is configured, this value will override the route preference.

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

Specifying Community Information

By default, no Border Gateway Protocol (BGP) community information is associated with static routes. To associate community information with IPv4 routes, include the community statement:

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

To associate community information with IPv6 routes, include the community statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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 be a value in the range from 1 through 65,534. *community-value* is the community identifier and can be a number in the range from 0 through 65,535.

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

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

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

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

You can 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.

The format for extended community identifiers is:

```
type:administrator:assigned-number
```

type is the type of extended community and can be a target, origin, or domain-id community. The target community identifies the destination to which the route is going. The origin community identifies where the route originated. The domain-id community identifies the OSPF domain where the route originated.

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

assigned-number identifies the local provider.

Specifying the AS Path

By default, no AS path information is associated with static routes. To associate AS path information with IPv4 routes, include the `as-path` statement:

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

To associate AS path information with IPv6 routes, include the `as-path` statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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 IP address of the BGP system that formed the static route.

Specifying the OSPF Tag

By default, no OSPF tag strings are associated with static routes. You can specify an OSPF tag string by including the tag statement:

```
[edit routing-options static (defaults | route)]
tag string;
```

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

Specifying Whether a 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 IPv4 static routes into the forwarding table, include the no-install statement:

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

To configure the software not to install active IPv6 static routes into the forwarding table, include the no-install statement:

```
[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 IPv4 routes into the forwarding table, include the install statement. Include this statement when configuring an individual route in the route portion of the static statement to override a no-install option specified in the defaults portion of the statement.

```
[edit routing-options static (defaults | route)]
install;
```

To explicitly install IPv6 routes into the forwarding table, include the install statement. 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;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying 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 an IPv4 static route remain in the forwarding table, include the retain statement. 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 static (defaults | route)]
retain;
```

To have an IPv6 static route remain in the forwarding table, include the `retain` statement. 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 IPv4 routes be deleted from the forwarding table, include the `no-retain` statement. Include this 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 static (defaults | route)]
no-retain;
```

To explicitly specify that IPv6 routes be deleted from the forwarding table, include the `no-retain` statement. 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;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table

Static routes are only removed from the routing table if the next hop becomes unreachable. This can occur if the local or neighbor interface goes down. To have an IPv4 static route remain installed in the routing and forwarding tables, include the passive statement:

```
[edit routing-options static (defaults | route)]
  passive;
```

To have an IPv6 static route remain installed in the routing and forwarding tables, include the passive statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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 IPv4 static routes when they become inactive, include the active statement. Include this 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 static (defaults | route)]
  active;
```

To explicitly remove IPv6 static routes when they become inactive, include the active statement. 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;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying When the Route Can Be Readvertised

By default, static routes are eligible to be readvertised (that is, exported) by dynamic routing protocols. To mark an IPv4 static route as being ineligible for readvertisement, include the no-readvertise statement:

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

To mark an IPv6 static route as being ineligible for readvertisement, include the no-readvertise statement:

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

To explicitly readvertise IPv4 static routes, include the `readvertise` statement. Include the `readvertise` 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 static (defaults | route)]
readvertise;
```

To explicitly readvertise IPv6 static routes, include the `readvertise` statement. 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;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying When the Route Can Be Resolved to a Prefix That Is Not Directly Connected

By default, static routes can point only to a directly connected next hop. You can configure an IPv4 route to a prefix that is not directly connected by resolving the route through the `inet.0` and `inet.3` routing tables. To configure an IPv4 static route to a prefix that is not a directly connected next hop, include the `resolve` statement:

```
[edit routing-options static (defaults | route)]
resolve;
```

You can configure an IPv6 route to a prefix that is not directly connected by resolving the route through the `inet6.0` routing table. To configure an IPv6 static route to a prefix that is not a directly connected next hop, include the `resolve` statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Configuring Bidirectional Forwarding Detection

The bidirectional forwarding detection (BFD) protocol is a simple hello mechanism that detects failures in a network. BFD works with a wide variety of network environments and topologies. The failure detection timers for BFD have shorter time limits than the failure detection mechanisms of IS-IS, providing faster detection. These timers are also adaptive and can be adjusted to be more or less aggressive.

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

```
[edit routing-options static (defaults | route)]
bfd-liveness-statement {
  minimum-interval milliseconds;
  minimum-receive-interval milliseconds;
  minimum-transmit-interval milliseconds;
  multiplier number;
}
```

To specify the minimum transmit and receive interval for failure detection, include the `minimum-interval` statement:

```
[edit routing-options static (defaults | route) bfd-liveness-statement]
minimum-interval milliseconds;
```

To specify the minimum receive interval for failure detection, include the `minimum-receive-interval` statement:

```
[edit routing-options static (defaults | route) bfd-liveness-statement]
minimum-receive-interval milliseconds;
```

To specify the minimum transmit interval for failure detection, include the `minimum-transmit-interval` statement:

```
[edit routing-options static (defaults | route) bfd-liveness-statement]
minimum-transmit-interval milliseconds;
```

To specify the detection time multiplier for failure detection, include the `multiplier` statement:

```
[edit routing-options static (defaults | route) bfd-liveness-statement]
multiplier number;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

To enable failure detection for IPv6 routes, include the `bfd-liveness-detection` statement:

```
[edit routing-options rib inet6.0 static (defaults | route)]
bfd-liveness-statement {
  minimum-interval milliseconds;
  minimum-receive-interval milliseconds;
  minimum-transmit-interval milliseconds;
  multiplier number;
}
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Configuring a Default Route

To configure an IPv4 default route, include the next-hop address and retain statements:

```
[edit routing-options static route default]
next-hop address;
retain;
```

To configure an IPv6 static route, include the next-hop address and retain statements:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Propagating 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 (commonly, 127.0.0.1). However, configuring static routes in this way with the JUNOS software (by including a statement such as `route address/mask-length next-hop 127.0.0.1`) does not propagate the static routes, because the forwarding table ignores static routes whose next-hop router is the loopback address. To propagate IPv4 static routes into the routing protocols, include the discard statement:

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

To propagate IPv6 static routes into the routing protocols, include the discard statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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

Examples: Configuring Static Routes

Configure an IPv4 default route through the next-hop router 192.238.52.33:

```
[edit]
user@host# set routing-options static route 0.0.0.0/0 next-hop 192.238.52.33
[edit]
user@host# show
routing-options {
  static {
    route 0.0.0.0/0 next-hop 192.238.52.33;
  }
}
```

Configure IPv4 static routes that are retained in the forwarding table when the routing software shuts down normally:

```
[edit]
user@host# set routing-options static route 0.0.0.0/0 next-hop 192.168.1.254
retain
[edit]
user@host# set routing-options static route 10.1.1.1/32 next-hop 127.0.0.1
retain
[edit]
user@host# show
routing-options {
  static {
    route 0.0.0.0/0 {
      next-hop 192.168.1.254;
      retain;
    }
    route 10.1.1.1/32 {
      next-hop 127.0.0.1;
      retain;
    }
  }
}
```

Configure an IPv4 static route and have it propagate into the routing protocols. In this example, do not specify the route as 143.172.0.0/6 next-hop 127.0.0.1:

```
[edit]
user@host# set routing-options static route 143.172.0.0/6 discard
[edit]
user@host# show
routing-options {
  static {
    route 143.172.0.0/6 discard;
  }
}
```

Install an IPv4 static route into both inet.0 and inet.2:

```
[edit]
user@host# set routing-options static rib-group some-group
user@host# set rib-groups some-group import-rib [inet.0 inet.2]
[edit]
user@host# show
routing-options {
  static {
    rib-group some-group;
  }
  rib-groups {
    some-group {
      import-rib [ inet.0 inet.2 ];
    }
  }
}
```

Configure an IPv6 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;
  }
}
```

Resolve an IPv6 static route to non-next-hop router 1::/64 using next-hop router 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
```

Configuring Aggregate Routes

Route aggregation allows you to combine groups of routes with common addresses into a single entry in the routing table. This decreases the size of the routing table as well as the number of route advertisements sent by the router.

An aggregate route becomes active when it has one or more *contributing routes*. A contributing route is an active route that is a more specific match for the aggregate destination. For example, for the aggregate destination 128.100.0.0/16, routes to 128.100.192.0/19 and 128.100.67.0/24 are contributing routes, but routes to 128.0.0.0/8, 128.0.0.0/16, and 128.100.0.0/16 are not.

A route can contribute only to a single aggregate route. However, an active aggregate route can recursively contribute to a less specific matching aggregate route. For example, an aggregate route to the destination 128.100.0.0/16 can contribute to an aggregate route to 128.96.0.0/13.

When an aggregate route becomes active, it is installed in the routing table with the following information:

Reject next hop—If a more-specific packet does not match a more-specific route, the packet is rejected and an ICMP unreachable message is sent to the packet's originator.

Metric value as configured with the aggregate statement.

Preference value that results from the policy filter on the primary contributor, if a filter is specified.

AS path as configured in the aggregate statement, if any. Otherwise, the path is computed by aggregating the paths of all contributing routes.

Community as configured in the aggregate statement, if any is specified.



NOTE: You can configure only one aggregate route for each destination prefix.

To configure aggregate routes in the default routing table (inet.0), include the aggregate statement:

```
[edit]
routing-options {
  aggregate {
    defaults {
      aggregate-options;
    }
    route destination-prefix {
      policy policy-name;
      aggregate-options;
    }
  }
}
```

To configure aggregate routes in one of the other routing tables, or to explicitly configure aggregate routes in the default routing table (inet.0), include the aggregate statement:

```
[edit]
routing-options {
  rib routing-table-name {
    aggregate {
      defaults {
        aggregate-options;
      }
      route destination-prefix {
        policy policy-name;
        aggregate-options;
      }
    }
  }
}
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The aggregate statement consists of two parts:

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

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

The following sections explain how to configure aggregate routes.

Specifying the Destination of the Aggregate Route on page 67

Specifying Aggregate Route Options on page 68

Specifying Policy with Aggregate Routes on page 73

Advertising Aggregate Routes on page 74

Specifying the Destination of the Aggregate Route

When you configure an individual aggregate route in the route part of the aggregate 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 IP address and *mask-length* is the destination prefix length.

default if this is the default route to the destination. This is equivalent to specifying an IP address of 0.0.0.0/0.

Specifying Aggregate Route Options

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

To configure aggregate route options, include one or more of them in the defaults or route part of the aggregate statement. For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement. Each of these options is explained in the sections that follow.

```
[edit]
routing-options {
  aggregate {
    (defaults | route) {
      (active | passive);
      as-path <as-path> <origin (egp | igp | incomplete)> <atomic-aggregate>
        <aggregator as-number in-address>;
      community [ community-ids ];
      discard;
      (brief | full);
      (metric | metric2 | metric3 | metric4)metric <type type>;
      (preference | preference2 | color | color2) preference <type type>;
      tag string;
    }
  }
}
```

The following sections explain how to specify aggregate route options:

Specifying the Route Metric on page 69

Specifying the Route Preference on page 69

Specifying a Next Hop for a Route on page 69

Specifying Community Information on page 70

Specifying the AS Path on page 71

Specifying Which AS Numbers to Include in the Aggregate Route on page 71

Specifying the OSPF Tag on page 72

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table on page 72

Specifying the Route Metric

You can specify up to four metric values, starting with `metric` (for the first metric value) and continuing with `metric2`, `metric3`, and `metric4` by including one or more of the following statements:

```
[edit routing-options aggregate (defaults | route)]
(metric | metric2 | metric3 | metric4) metric <type type>;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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

Specifying the Route Preference

By default, aggregate routes have a preference value of 130. If the routing table contains a dynamic route to a destination that has a better (lower) preference value than this, the dynamic route is chosen as the active route and is installed in the forwarding table.

To modify the default preference value, specify a primary preference value (`preference`). You also can specify 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:

```
[edit routing-options aggregate (defaults | route)]
(preference | preference2 | color | color2) preference <type type>;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The preference value can be a number in the range from 1 through 255, with a lower number indicating a more preferred route. For more information about preference values, see “Route Preferences” on page 6.

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

Specifying a Next Hop for a Route

By default, when aggregate routes are installed in the routing table, the next hop is configured as a reject route. That is, the packet is rejected and an ICMP unreachable message is sent to the packet’s originator.

When you configure an individual route in the `route` part of the aggregate statement, or when you configure the defaults for aggregate routes, you can specify a discard next hop. This means that if a more specific packet does not match a more specific route, the packet is rejected and a reject route for this destination is installed in the routing table, but ICMP unreachable messages are not sent. Being able to discard next hops allows you to originate a summary route, which is advertisable through dynamic routing protocols, and allows you to discard received traffic that does not match a more specific route than the summary route. To discard next hops, include the `discard` option:

```
[edit routing-options aggregate defaults]
discard;
```

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

Specifying Community Information

By default, no BGP community information is associated with aggregate routes. To associate community information with the routes, include the community option:

```
[edit routing-options aggregate (defaults | route)]
  community [ community-ids ];
```

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

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 AS number and can be a value in the range from 1 through 65,534. *community-value* is the community identifier and can be a number in the range from 0 through 65,535.

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

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

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

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

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

The format for extended community identifiers is:

```
type:administrator:assigned-number
```

type is the type of extended community and can be a target, origin, or domain-id community. The target community identifies the destination to which the route is going. The origin community identifies where the route originated. The domain-id community identifies the OSPF domain where the route originated.

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

assigned-number identifies the local provider.

Specifying the AS Path

By default, the AS path for aggregate routes is built from the component routes. To manually specify the AS path and associate AS path information with the routes, include the `as-path` option:

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

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

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:

`egp`—Path information originated in another AS.

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

`incomplete`—Path information was learned by some other means.

To attach the BGP `ATOMIC_AGGREGATE` path attribute to the aggregate route, specify the `atomic-aggregate` option. 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 aggregate route, specify the `aggregator` option. When using this option, you must specify the last AS number that formed the aggregate route (encoded as two octets), followed by the IP address of the BGP system that formed the aggregate route.

Specifying Which AS Numbers to Include in the Aggregate Route

By default, all AS numbers from all contributing paths are included in the aggregate route's path. To include only the longest common leading sequences from the contributing AS paths, include the `brief` option when configuring the route. If doing this results in AS numbers being omitted from the aggregate route, the BGP `ATOMIC_ATTRIBUTE` path attribute is included with the aggregate route.

```
[edit routing-options aggregate (defaults | route)]
  brief;
```

To explicitly have all AS numbers from all contributing paths be included in the aggregate route's path, include the `full` option when configuring routes. Include this option when configuring an individual route in the route portion of the aggregate statement to override a `retain` option specified in the defaults portion of the statement.

```
[edit routing-options aggregate (defaults | route)]
full;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying the OSPF Tag

By default, no OSPF tag strings are associated with aggregate routes. You can specify an OSPF tag string by including the `tag` option:

```
[edit routing-options aggregate (defaults | route)]
tag string;
```

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

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table

Static routes are only removed from the routing table if the next hop becomes unreachable. The next hop will become unreachable if there are no contributing routes. To have an aggregate route remain continually installed in the routing and forwarding tables, include the `passive` option when configuring the route:

```
[edit routing-options aggregate (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 aggregate routes when they become inactive, include the `active` option when configuring routes. Include this option when configuring an individual route in the route portion of the aggregate statement to override a `retain` option specified in the defaults portion of the statement.

```
[edit routing-options aggregate (defaults | route)]
active;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying Policy with Aggregate Routes

You can associate a routing policy when configuring an aggregate route's destination prefix in the routes part of the aggregate statement. Doing so provides the equivalent of an import routing policy filter for the destination prefix. That is, each potential contributor to an aggregate route, along with any aggregate options, is passed through the policy filter. The policy then can accept or reject the route as a contributor to the aggregate route and, if the contributor is accepted, the policy can modify the default preferences.

The following algorithm is used to compare two aggregate contributing routes in order to determine which one is the primary or preferred contributor:

1. Compare the protocol's preferences of the contributing routes. The lower the preference, the better the route. This is similar to the comparison that is done while determining the best route for the routing table.
2. Compare the protocol's preference2 of the contributing routes. The lower preference2 value is better. If only one route has preference2, then this route is preferred.
3. The preference values are the same. Proceed with a numerical comparison of the prefix values.
 - a. The primary contributor is the numerically smallest prefix value.
 - b. If the two prefixes are numerically equal, the primary contributor is the route that has the smallest prefix length value.
4. At this point, the two routes are the same. The primary contributor does not change. An additional next hop will be available for the existing primary contributor.

A rejected contributor still can contribute to a less specific aggregate route. If you do not specify a policy filter, all candidate routes contribute to an aggregate route.

To associate a routing policy with an aggregate route, include the policy statement when configuring the route:

```
[edit routing-options aggregate (defaults | route)]
  policy policy-name;
```

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

Advertising Aggregate Routes

After you have configured aggregate routes, you can have a protocol advertise the routes by configuring a policy that is then exported by a routing protocol.

To configure a protocol to advertise routes, include the policy-statement statement:

```
[edit]
policy-options {
  policy-statement advertise-aggregate-routes {
    term first-term {
      from protocol aggregate;
      then accept;
    }
    term second-term {
      then next policy;
    }
  }
}
protocols {
  bgp {
    export advertise-aggregate-routes;
    ...
  }
}
```

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

Configuring Generated Routes

Generated routes are used as the *route of last resort*. A packet is forwarded to the route of last resort when the routing tables have no information about how to reach that packet's destination. One use of route generation is to generate a default route to use if the routing table contains a route from a peer on a neighboring backbone.

A generated route becomes active when it has one or more *contributing routes*. A contributing route is an active route that is a more specific match for the generated destination. For example, for the destination 128.100.0.0/16, routes to 128.100.192.0/19 and 128.100.67.0/24 are contributing routes, but routes to 128.0.0.0/8, 128.0.0.0/16, and 128.100.0.0/16 are not.

A route can contribute only to a single generated route. However, an active generated route can recursively contribute to a less specific matching generated route. For example, a generated route to the destination 128.100.0.0/16 can contribute to a generated route to 128.96.0.0/13.

By default, when generated routes are installed in the routing table, the next hop is chosen from the primary contributing route.



NOTE: Currently, you can configure only one generated route for each destination prefix.

To configure generated routes in the default routing table (inet.0), include the generate statement:

```
[edit]
routing-options {
  generate {
    defaults {
      generate-options;
    }
    route destination-prefix {
      policy policy-name;
      generate-options;
    }
  }
}
```

To configure generated routes in one of the other routing tables, or to explicitly configure generated routes in the default route table (inet.0), include the generate statement:

```
[edit]
routing-options {
  rib routing-table-name {
    generate {
      defaults {
        generate-options;
      }
      route destination-prefix {
        policy policy-name;
        generate-options;
      }
    }
  }
}
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The generate statement consists of two parts:

defaults—Here you specify global generated route options. These are treated as global defaults and apply to all the generated routes you configure in the generate statement. This part of the generate statement is optional.

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

The following sections explain how to configure generated routes.

Specifying the Destination of a Generated Route on page 76

Specifying Generated Route Options on page 76

Specifying Policy with Generated Routes on page 81

Specifying the Destination of a Generated Route

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

network/masklen, where *network* is the network portion of the IP address and *masklen* is the destination prefix length.

default if this is the default route to the destination. This is equivalent to specifying an IP address of 0.0.0.0/0.

Specifying Generated Route Options

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

To configure generated route options, include one or more of them in the defaults or route part of the generate statement (for routing instances, include the statement). For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement. Each of these options is explained in the sections that follow.

```
[edit]
routing-options {
  generate {
    (defaults | route) {
      (active | passive);
      as-path <as-path> <origin (egp | igp | incomplete)> <atomic-aggregate>
        <aggregator as-number in-address>;
      community [ community-ids ];
      discard;
      (brief | full);
      (metric | metric2 | metric3 | metric4) metric <type type>;
      (preference | preference2 | color | color2) preference <type type>;
      tag string;
    }
  }
}
```

The following sections explain how to specify generated route options:

Specifying the Route Metric on page 77

Specifying the Route Preference on page 77

Specifying a Next Hop for a Route on page 77

Specifying Community Information on page 78

Specifying the AS Path on page 79

Specifying the OSPF Tag on page 80

Specifying Which AS Numbers to Include in the Generated Route on page 80

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table on page 80

Specifying the Route Metric

You can specify up to four metric values, starting with `metric` (for the first metric value) and continuing with `metric2`, `metric3`, and `metric4` by including one or more of the following statements:

```
[edit routing-options generate (defaults | route)]
(metric | metric2 | metric3 | metric4) metric <type type>;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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

Specifying the Route Preference

By default, generated routes have a preference value of 130. If the JUNOS routing table contains a dynamic route to a destination that has a better (lower) preference value than this, the dynamic route is chosen as the active route and is installed in the forwarding table.

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:

```
[edit routing-options generate (defaults | route)]
(preference | preference2 | color | color2) preference <type type>;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

The preference value can be a number in the range from 1 through 255, with a lower number indicating a more preferred route. For more information about preference values, see “Route Preferences” on page 6.

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

Specifying a Next Hop for a Route

By default, when generated routes are installed in the routing table, the next hop is chosen from the primary contributing route.

When you configure an individual route in the route part of the generate statement, or when you configure the defaults for generated routes, you can specify a discard next hop. This means that if a more specific packet does not match a more specific route, the packet is rejected and a reject route for this destination is installed in the routing table, but ICMP unreachable messages are not sent. The discard next-hop feature allows you to originate a summary route, which is advertisable through dynamic routing protocols, and allows you to discard received traffic that does not match a more specific route than the summary route.

For example:

```
[edit routing-options generate route 1.0.0.0/8]
user@host# set discard
```

Specifying Community Information

By default, no BGP community information is associated with generated routes. To associate community information with the routes, include the community option:

```
[edit routing-options generate (defaults | route)]
community [ community-ids ];
```

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

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 AS number and can be a value in the range from 1 through 65,534. *community-value* is the community identifier and can be a number in the range from 0 through 65,535.

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

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 explicitly exclude BGP community information with a generated route using the none option. Include none when configuring an individual route in the route portion of the generate statement to override a community option specified in the defaults portion of the statement.

The format for extended community identifiers is:

```
type:administrator:assigned-number
```

type is the type of extended community and can be a target, origin, or domain-id community. The target community identifies the destination to which the route is going. The origin community identifies where the route originated. The domain-id community identifies the OSPF domain where the route originated.

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

assigned-number identifies the local provider.

Specifying the AS Path

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

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

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

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:

egp—Path information originated in another AS.

igp—Path information originated within the local AS.

incomplete—Path information was learned by some other means.

To attach the BGP ATOMIC_AGGREGATE path attribute to the generated route, specify the *atomic-aggregate* option. 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 generated route, specify the *aggregator* option. When using this option, you must specify the last AS number that formed the generated route (encoded as two octets), followed by the IP address of the BGP system that formed the generated route.

Specifying the OSPF Tag

By default, no OSPF tag strings are associated with generated routes. You can specify an OSPF tag string by including the tag option:

```
[edit routing-options generate (defaults | route)]
tag string;
```

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

Specifying Which AS Numbers to Include in the Generated Route

By default, all AS numbers from all contributing paths are included in the generated route's path. To include only the longest common leading sequences from the contributing AS paths, include the brief option when configuring the route. If doing this results in AS numbers being omitted from the generated route, the BGP ATOMIC_ATTRIBUTE path attribute is included with the generated route.

```
[edit routing-options generate (defaults | route)]
brief;
```

To explicitly have all AS numbers from all contributing paths be included in the generated route's path, include the full option when configuring routes. Include this option when configuring an individual route in the route portion of the generate statement to override a retain option specified in the defaults portion of the statement.

```
[edit routing-options generate (defaults | route)]
full;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying Whether Inactive Routes Are Removed from the Routing or Forwarding Table

Static routes are only removed from the routing table if the next hop becomes unreachable. The next hop will become unreachable if there are no contributing routes. To have a generated route remain continually installed in the routing and forwarding tables, include the passive option when configuring the route:

```
[edit routing-options generate (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 generated routes when they become inactive, include the active option when configuring routes. Include this option when configuring an individual route in the route portion of the generate statement to override a retain option specified in the defaults portion of the statement.

```
[edit routing-options generate (defaults | route)]
active;
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Specifying Policy with Generated Routes

You optionally can associate a routing policy when configuring a generated route's destination prefix in the routes part of the generate statement. Doing so provides the equivalent of an import routing policy filter for the destination prefix. That is, each potential contributor to a generated route, along with any generate options, is passed through the policy filter. The policy can accept or reject the route as a contributor to the generated route and, if the contributor is accepted, the policy can modify the default preferences.

The following algorithm is used to compare two generated contributing routes in order to determine which one is the primary or preferred contributor:

1. Compare the protocol's preference of the contributing routes. The lower the preference, the better the route. This is similar to the comparison that is done while determining the best route for the routing table.
2. Compare the protocol's preference2 of the contributing routes. The lower preference2 value is better. If only one route has preference2, then this route is preferred.
3. The preference values are the same. Proceed with a numerical comparison of the prefixes values.
 - a. The primary contributor is the numerically smallest prefix value.
 - b. If the two prefixes are numerically equal, the primary contributor is the route that has the smallest prefix length value.

At this point, the two routes are the same. The primary contributor does not change. An additional next hop will be available for the existing primary contributor.

A rejected contributor still can contribute to less specific generated route. If you do not specify a policy filter, all candidate routes contribute to a generated route.

To associate a routing policy with an generated route, include the policy statement:

```
[edit routing-options generate route]
policy policy-name;
```

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

Configuring 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 IPv4, the following are the default martian addresses:

```
0.0.0.0/8
127.0.0.0/8
128.0.0.0/16
191.255.0.0/16
192.0.0.0/24
223.255.255.0/24
240.0.0.0/4
```

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

The following sections explain how to configure martian routes:

Adding Martian Addresses on page 82

Deleting Martian Addresses on page 83

Adding Martian Addresses

To add martian addresses to the list of default martian addresses in the default IPv4 routing table (inet.0), include the martians statement:

```
[edit]
routing-options {
  martians {
    destination-prefix match-type;
  }
}
```

To add martian addresses to the list of default martian addresses in 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:

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

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 default routing table (inet.0), include the martians statement:

```
[edit]
routing-options {
  rib routing-table-name {
    martians {
      destination-prefix match-type;
    }
  }
}
```

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

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

default—If this is the default route to the destination. This is equivalent to specifying the IP address 0.0.0.0/0.

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

In *match-type*, specify the type of match to apply to the destination prefix. For more information about match types, see the *JUNOS Policy Framework Configuration Guide*.

Deleting Martian Addresses

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

To delete a martian address from the default routing table (inet.0), include the martians statement:

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

To delete a martian address from other routing tables, or to explicitly delete a martian address from the primary IPv6 routing table (inet6.0), include the martians statement:

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

For a list of hierarchy levels at which you can configure these statements, see the statement summary sections for these statements.

Applying a Filter to a Forwarding Table

To apply an input filter to a forwarding table, include the input statement:

```
[edit]
routing-options {
  rib routing-table-name {
    filter {
      input filter-name;
    }
  }
}
```

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



NOTE: Forwarding table filtering is not supported on the interfaces you configure as tunnel sources. This affects only the transit packets exiting the tunnel.

For more information about forwarding table filters, see the *JUNOS Policy Framework Configuration Guide*.