

Chapter 5

Configuring Other Protocol-Independent Routing Properties

This chapter discusses how to perform the following tasks for configuring other protocol-independent routing properties:

Configuring the AS Number on page 84

Configuring the Router Identifier on page 84

Configuring AS Confederation Members on page 84

Configuring Route Recording for Flow Aggregation on page 85

Creating Routing Table Groups on page 86

Configuring How Interface Routes Are Imported into Routing Tables on page 87

Configuring Multicast Scoping on page 89

Configuring Additional Source-Specific Multicast Groups on page 90

Configuring Per-Packet Load Balancing on page 90

Enabling Unicast Reverse-Path Forwarding Check on page 93

Configuring Graceful Restart on page 93

Configuring a Route Distinguisher on page 94

Configuring a Dynamic Tunnel on page 94

Configuring Logging for the Routing Protocol Process on page 95

Tracing Global Routing Protocol Operations on page 96

Configuring the AS Number

An autonomous system (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 the Border Gateway Protocol (BGP) on the router, you must configure an AS number.

To configure the router's AS number, include the `autonomous-system` statement:

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

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

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

Configuring the Router Identifier

The router identifier is used by BGP and Open Shortest Path First (OSPF) to identify the router from which a packet originated. The router identifier usually is the IP address of the local router. If you do not configure a router identifier, the IP address of the first interface to come online is used. This is usually the loopback interface. Otherwise, the first hardware interface with an IP address is used.

To configure the router identifier, include the `router-id` statement:

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

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

Configuring 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:

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

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

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.

Configuring Route Recording for Flow Aggregation

Before you can perform flow aggregation, the routing protocol process must export the AS path and routing information to the sampling process. To do this, include the route-record statement:

```
[edit]
routing-options {
  route-record;
}
```

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

For more information about flow aggregation and sampling, see the *JUNOS Network Interfaces and Class of Service Configuration Guide*.

Creating 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:

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

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

The routing table group can have any name you choose (specified in *group-name*). If the group name you specify is not created explicitly, as described in “Configuring Other Protocol-Independent Routing Properties” on page 83, you can create it by naming it in the `rib-groups` statement.

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 the *secondary routing tables*.

The primary routing table determines the address family of the routing table group. To configure an Internet Protocol version 4 (IPv4) routing table group, specify `inet.0` as the primary routing table. To configure an Internet Protocol version 6 (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).

If you have configured a routing table, configure the OSPF primary instance at the `[edit protocols ospf]` hierarchy level with the statements needed for your network so that routes are installed in `inet.0` and in the forwarding table. Make sure to include the routing table group. For more information, see “Configuring Multiple Instances of OSPF” on page 172.

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:

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

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

Examples: Creating Routing Table Groups

Create an IPv4 routing table group so that interface routes are installed into two routing tables, `inet.0` and `inet.2`:

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

Create an IPv6 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 {
    import-rib [ inet6.0 inet6.2 ];
  }
}
```

Configuring How Interface Routes Are Imported into Routing Tables

By default, IPv4 interface routes (also called direct routes) are imported into routing table `inet.0`, and IPv6 interface routes are imported into routing table `inet6.0`. If you are configuring alternate routing tables for use by some routing protocols, it might be necessary to import the interface routes into the alternate routing tables. To define the routing tables into which interface routes are imported, you create a routing table group and associate it with the router's interfaces.

To associate an IPv4 routing table group with the router's interfaces and specify which routing table groups interface routes are imported into, include the `interface-routes` statement:

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

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

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

To create the routing table groups, include the `rib-groups` statement at the `[edit routing-options]` hierarchy level. For configuration information, see “Creating Routing Table Groups” on page 86.

If you have configured a routing table, configure the OSPF primary instance at the `[edit protocols ospf]` hierarchy level with the statements needed for your network so that routes are installed in `inet.0` and in the forwarding table. Make sure to include the routing table group. For more information, see “Configuring Multiple Instances of OSPF” on page 172.

To export local routes, include the `export` statement:

```
[edit]
routing-options {
  interface-routes {
    family (inet | inet6) {
      export {
        lan;
        point-to-point;
      }
    }
  }
}
```

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

To export lan routes, include the `lan` option. To export point-to-point routes, include the `point-to-point` option.

Only local routes on point-to-point interfaces configured with a destination address are exportable.

Configuring Multicast Scoping

To configure multicast address scoping, include the following statements:

```
[edit]
routing-options {
  multicast {
    scope scope-name {
      interface [ interface-names ];
      prefix destination-prefix;
    }
  }
}
```

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

Specify a name for the scope, its address range, and the router interfaces on which you are configuring scoping.

Example: Configuring Multicast Scoping

Configure multicast scoping by creating four scopes: local, organization, engineering, and marketing.

Configure the local scope on a Fast Ethernet interface. Configure the organization scope on a Fast Ethernet and a SONET/SDH interface. Configure the engineering and marketing scopes on two SONET/SDH interfaces.

```
[edit]
routing-options {
  multicast {
    scope local {
      prefix 239.255.0.0/16;
      fe-0/1/0.0;
    }
    scope organization {
      prefix 239.192.0.0/14;
      interface [ fe-0/1/0.0 so-0/0/0.0 ];
    }
    scope engineering {
      prefix 239.255.255.0/24;
      interface [ so-0/0/1.0 so-0/0/2.0 ];
    }
    scope marketing {
      prefix 239.255.254.0/24;
      interface [ so-0/0/1.0 so-0/0/2.0 ];
    }
  }
}
```

Configuring Additional Source-Specific Multicast Groups

IGMPv3 supports Source Specific Multicast (SSM) groups. By utilizing inclusion lists, only sources that are specified send to the SSM group. By default, the SSM group multicast address is limited to the IP address range 232.0.0.0 to 232.255.255.255. You can configure additional SSM groups. Shared tree delivery is prohibited on SSM groups.

To configure additional SSM groups, include the `ssm-groups` statement:

```
[edit]
routing-options {
  multicast {
    ssm-groups {
      address;
    }
  }
}
```

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

Configuring Per-Packet Load Balancing

For the active route, when there are multiple equal-cost paths to the same destination, by default, the JUNOS software chooses in a random fashion one of the next-hop addresses to install into the forwarding table. Whenever the set of next hops for a destination changes in any way, the next-hop address is chosen again, also in a random fashion.

You can configure the JUNOS software so that, for the active route, all next-hop addresses for a destination are installed in the forwarding table. This is called per-packet load balancing. You can use load balancing to spread traffic across multiple paths between routers. The behavior of the per-packet load-balancing function varies according to the version of the Internet Protocol ASIC in the router.

On routers with an Internet Processor ASIC, when per-packet load balancing is configured, traffic between routers with multiple paths is spread in a random fashion across the available interfaces. The forwarding table balances the traffic headed to a destination, transmitting packets in round-robin fashion among the multiple next hops (up to a maximum of eight equal-cost load-balanced paths). The traffic is load-balanced on a per-packet basis.



NOTE: Per-packet load distribution uses a hashing algorithm that distributes packets over equal-cost links. The algorithm is designed to distribute packets to prevent any single link from being saturated. However, per-packet load balancing offers no guarantee of equal distribution of traffic over equal-cost links, nor does it guarantee that increasing the number of Internet flows will create a better hash distribution.

On routers with the Internet Processor II ASIC, when per-packet load balancing is configured, traffic between routers with multiple paths is divided into individual traffic flows (up to a maximum of 16 equal-cost load-balanced paths). Packets for each individual flow are kept on a single interface. To recognize individual flows in the transit traffic, the router examines each of the following:

- Source IP address
- Destination IP address
- Protocol
- Source port number
- Destination port number
- Source interface index
- Type of service (ToS)

The router recognizes packets in which all of these parameters are identical, and it ensures that these packets are sent out through the same interface. This prevents problems that might otherwise occur with packets arriving at their destination out of their original sequence.

The following steps show how to configure per-packet load balancing:

1. Define a load-balancing routing policy by including one or more policy-statement statements at the [edit policy-options] hierarchy level, defining an action of load-balance per-packet:

```
[edit]
policy-options {
  policy-statement policy-name {
    from {
      match-conditions;
      route-filter destination-prefix match-type <actions>;
      prefix-list name;
    }
    then {
      load-balance per-packet;
    }
  }
}
```

2. Apply the policy to routes exported from the routing table to the forwarding table. To do this, include the export statement:

```
[edit]
routing-options {
  forwarding-table {
    export policy-name;
  }
}
```



NOTE: You cannot apply the export policy to VRF routing instances.

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



NOTE: Specify all next-hops of that route, if more than one exists, when allocating a label corresponding to a route that is being advertised.



NOTE: Configure the forwarding-options hash key for MPLS to include the IP payload.

Examples: Configuring Per-Packet Load Balancing

Perform per-packet load balancing for all routes:

```
[edit]
policy-options {
  policy statement load-balancing-policy {
    then {
      load-balance per-packet;
    }
  }
}
routing-options {
  forwarding-table {
    export load-balancing-policy;
  }
}
```

Perform per-packet load balancing for only a limited set of routes:

```
[edit]
policy-options {
  policy-statement load-balancing-policy {
    from {
      route-filter 192.168.10/24 orlonger;
      route-filter 9.114/16 orlonger;
    }
    then {
      load-balance per-packet;
    }
  }
}
routing-options {
  forwarding-table {
    export load-balancing-policy;
  }
}
```

Enabling Unicast Reverse-Path Forwarding Check

IP spoofing may occur during a denial-of-service (DoS) attack. IP spoofing allows an intruder to pass IP packets to a destination as genuine traffic, when in fact the packets are not actually meant for the destination. This type of spoofing is harmful because it consumes the destination's resources.

Unicast reverse-path forwarding (RPF) check is a tool to reduce forwarding of IP packets that may be spoofing an address. A unicast RPF check performs a route table lookup on an IP packet's source address, and checks the incoming interface. The router determines whether the packet is arriving from a path that the sender would use to reach the destination. If the packet is from a valid path, the router forwards the packet to the destination address. If it is not from a valid path, the router discards the packet. Unicast RPF is supported for the IPv4 and IPv6 protocol families, as well as for the virtual private network (VPN) address family.

To enable unicast RPF check, include the unicast-reverse-path statement:

```
[edit]
routing-options {
  forwarding-table {
    unicast-reverse-path (active-paths | feasible-paths);
  }
}
```

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

To consider only active paths during the unicast RPF check, include the active-paths option. To consider all feasible paths during the unicast RPF check, include the feasible-paths option.



NOTE: Reverse-path forwarding is not supported on the interfaces you configure as tunnel sources. This affects only the transit packets exiting the tunnel.

For more information about configuring unicast RPF, see the *JUNOS Network Interfaces and Class of Service Configuration Guide*.

Configuring Graceful Restart

Graceful restart allows a router undergoing a restart to inform its adjacent neighbors and peers of its condition. The restarting router requests a grace period from the neighbor or peer, which can then cooperate with the restarting router. With a graceful restart, the restarting router can still forward traffic during the restart period, and convergence in the network is not disrupted. The restart is not visible to the rest of the network, and the restarting router is not removed from the network topology.

The graceful restart request occurs only if the following conditions are met:

- The network topology is stable.

- The neighbor or peer cooperates.

The restarting router is not already cooperating with another restart already in progress.

The grace period does not expire.

Graceful restart is disabled by default. You must configure graceful restart at the [edit routing-options] hierarchy level to enable the feature for Layer 2 and Layer 3 VPNs.

To enable graceful restart, include the graceful-restart statement:

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

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

Configuring a Route Distinguisher

If the route distinguisher ID is configured, the routing process automatically generates a type 1 route distinguisher for VPN routing and forwarding (VRF) and Layer 2 VPN instances. If a route distinguisher is explicitly configured under the routing instances stanza, then that configured route distinguisher will be used.

To configure a route distinguisher identifier globally, include the route-distinguisher-id statement:

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

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

For more information about VPNs, see the *JUNOS VPNs Configuration Guide*.

Configuring a Dynamic Tunnel

A VPN that travels through a non-MPLS network requires a generic routing encapsulation (GRE) tunnel. This tunnel can be either a static tunnel or a dynamic tunnel. A static tunnel is configured manually between two provider edge (PE) routers. A dynamic tunnel is configured using BGP route resolution.

When a router receives a VPN route that resolves over a BGP next hop that does not have an MPLS path, a GRE tunnel can be created dynamically, allowing the VPN traffic to be forwarded to that route. Formerly, GRE tunnels had to be established manually. Only GRE IPv4 tunnels are supported.

To configure a dynamic tunnel between two PE routers, include the `dynamic-tunnel` statement:

```
[edit]
dynamic-tunnels tunnel-name {
  destination-prefix prefix;
  source-address address;
  tunnel-type type-of-tunnel;
}
```

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

Configuring Logging for the Routing Protocol Process

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

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

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

Include the following form of the statement to log messages for a particular severity level and all 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;
}
```

Make sure to include logging for the daemon facility. Additionally, syslog deals with processes logged at the info or notice severity level, so make sure that your regular syslog configurations include the info or notice levels.

Examples: Configuring 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;
}
```

Tracing 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 about tracing and the precedence of multiple tracing operations, see the *JUNOS System Basics Configuration Guide*.

To configure global routing protocol tracing flags, include the traceoptions statement:

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

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

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.



NOTE: Use the traceoption flags detail and all with caution. These flags may cause the CPU to become very busy.

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: Tracing Global Routing Protocol 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;
  }
}
```