

# Chapter 11

## RSVP Configuration Guidelines

To configure the Resource Reservation Protocol (RSVP), you include statements at the [edit protocols rsvp] hierarchy level of the configuration.

```
protocols {
  rsvp {
    disable;
    keep-multiplier number;
    preemption ( aggressive | disabled | normal );
    refresh-time seconds;
    traceoptions {
      file filename <replace> <size size> <files number> <no-stamp>
        <(world-readable | no-world-readable)>;
      flag flag <flag-modifier> <disable>;
    }
    interface interface-name {
      disable;
      (aggregate | no-aggregate);
      authentication-key key;
      bandwidth bps;
      hello-interval seconds;
      link-protection {
        bandwidth bandwidth;
        class-of-service class-of-service-value;
        disable;
      }
      subscription percentage;
      update-threshold threshold;
    }
  }
}
```

By default, RSVP is disabled.

This chapter describes the minimum required configuration and discusses the following tasks for configuring RSVP:

Minimum RSVP Configuration on page 138

Enable RSVP on page 138

Configure RSVP Aggregation on page 139

Enable RSVP Fast Reroute on page 139

Configure the RSVP Hello Interval on page 140

- Configure RSVP Authentication on page 140
- Reserve Bandwidth on an Interface on page 141
- Configure the RSVP Update Threshold on an Interface on page 141
- Configure RSVP Timers on page 142
- Preempt RSVP Sessions on page 143
- Trace RSVP Protocol Traffic on page 143
- Configure RSVP and MPLS on page 145

## Minimum RSVP Configuration

To enable RSVP on all interfaces, include the following statement in the configuration file. All other RSVP configuration statements are optional.

```
[edit]
protocols {
  rsvp {
    interface all;
  }
}
```

## Enable RSVP

To enable RSVP, including the following statements at the [edit] hierarchy level:

```
[edit]
protocols {
  rsvp {
    interface interface-name;
  }
}
```

To enable RSVP on all interfaces, specify *all* for *interface-name*.

If you have configured interface properties on a group of interfaces and want to disable RSVP on one of the interfaces, include the disable statement within the rsvp interface statement:

```
[edit]
protocols {
  rsvp {
    interface interface-name {
      disable;
    }
  }
}
```

## Configure RSVP Aggregation

The resource requirements—processing, bandwidth, and memory—for running RSVP on a router increase proportionally with the number of sessions. Handling large numbers of refresh messages transmitted between RSVP neighbors is crucial for supporting large numbers of sessions. Path and Resv messages typically represent the majority of refreshes.

If topology failures occur, every node adjacent to the failure might notify all affected sender and receiver nodes. These notification messages are either tear or error messages, and they typically represent a flood that ripples out from the original failure point.

Aggregation provides a mechanism for reducing message flooding and network overload. It also enhances the efficiency and reliability in delivering RSVP tear or error messages. Note that RSVP aggregation is called *Bundle Message* in the internet draft *RSVP Refresh Reduction Extensions*.

By default, aggregation is disabled on all interfaces. For interoperability with other routers, you might need to keep aggregation disabled. However, we recommend that you enable aggregation between Juniper Networks routers to improve scalability. If you have several thousand MPLS LSPs, you must enable aggregation to ensure stable operation. To enable aggregation, include the aggregate statement at the [edit protocols rsvp interface *interface-name*] hierarchy level:

```
[edit protocols rsvp interface interface-name]  
aggregate;
```

## Enable RSVP Fast Reroute

RSVP fast reroute allows you to configure a network to quickly reroute traffic around a single link. For each label-switching router (LSR), you configure a bypass label-switched path (LSP) for each RSVP neighbor. Each bypass LSP monitors all of the LSPs associated with the neighbor. If a neighbor fails, the LSPs are rerouted by means of the bypass LSP.

A bypass LSP cannot share the same egress interface with the LSPs it monitors. When the interface used by a bypass LSP fails, a new interface is needed, so all LSPs protected by the bypass LSP are switched to a new bypass LSP.

Enable RSVP fast reroute on an LSP by including the link-protection statement at [edit protocols mpls lsp *lsp-name*] hierarchy level:

```
[edit protocols mpls lsp lsp-name]  
link-protection {  
  bandwidth bandwidth;  
  class-of-service class-of-service-value;  
  disable;  
}
```

Enable RSVP fast reroute on an interface by including the link-protection statement at the [edit protocols rsvp interface *interface-name*] hierarchy level and specifying values for bandwidth and class-of-service:

```
[edit protocols rsvp interface interface-name]  
link-protection {  
  bandwidth bandwidth;  
  class-of-service class-of-service-value;  
  disable;  
}
```

## Configure the RSVP Hello Interval

RSVP hello packets enable RSVP nodes to detect the loss of a neighboring node's RSVP state information. (Losses typically occur when the neighboring router restarts or the link fails.) In standard RSVP, such detection occurs as a consequence of RSVP's soft-state model. However, detection typically requires several minutes to time out the soft state. RSVP hello packets detect the neighboring node's state changes much more quickly, usually within 10–20 seconds.

Between hello-capable neighbors, hello packets are sent unicast toward each other. A loss of  $(2 \times \text{keep-multiplier} + 1)$  consecutive hello packets causes the neighbor's state to go down, and all RSVP sessions to and from that neighbor are declared to be down.

JUNOS RSVP hello packets are optional and are backwards compatible with RSVP implementations that do not support hello packets. For neighbors that do not support hello packets, RSVP uses the soft-state timeout for loss detection.

If all neighboring nodes support hello packets, you can reduce the refresh overhead (by increasing the value set in the refresh-time statement) without adversely affecting the node or link failure detection time. Also, the network can scale to a larger number of sessions because the refresh operations consume less CPU and bandwidth. For information about setting the refresh overhead, see "Configure RSVP Timers" on page 142.

By default, RSVP sends hello packets every 3 seconds. To modify how often RSVP sends hello packets, include the hello-interval statement at the [edit protocols rsvp interface *interface-name*] hierarchy level:

```
[edit protocols rsvp interface interface-name]
hello-interval seconds;
```

## Configure RSVP Authentication

All RSVP protocol exchanges can be authenticated to guarantee that only trusted neighbors participate in setting up reservations. By default, RSVP authentication is disabled.

RSVP authentication uses an HMAC-MD5 message-based digest. This scheme produces a message digest based on a secret authentication key and the message contents. (The message contents also include a sequence number.) The computed digest is transmitted with RSVP messages. Once you have configured authentication, all received and transmitted RSVP messages with all neighbors are authenticated on this interface.

MD5 authentication also provides protection against forgery and message modification. However, it does not provide confidentiality because all messages are sent in clear text, and it does not prevent replay attacks.

By default, authentication is disabled. To enable authentication, configure a key on each interface by including the authentication-key statement at the [edit protocols rsvp interface *interface-name*] hierarchy level:

```
[edit protocols rsvp interface interface-name]
authentication-key key;
```

## Reserve Bandwidth on an Interface

For each interface on which RSVP is enabled, by default, RSVP permits all the interface's bandwidth (100 percent) to be used for RSVP reservations.

Oversubscription on an interface occurs when the aggregate demand of all RSVP sessions is allowed to exceed physical capacity of the link. You can use oversubscription to take advantage of the statistical nature of traffic patterns and to permit higher utilization of links. In particular, you can use oversubscription in places where peak utilizations of traffic do not coincide in time.

Undersubscription on an interface occurs when the total demand of all RSVP sessions is always less than the physical capacity of the link. You can use undersubscription to bound utilization of links and reduce congestion.

You can modify the link bandwidth used for RSVP reservations, either decreasing it below 100 percent or oversubscribing the interface. To do this, include the subscription statement at the [edit protocols rsvp interface *interface-name*] hierarchy level:

```
[edit protocols rsvp interface interface-name]  
subscription percentage;
```

*percentage* is the percentage of the interface's bandwidth that RSVP allows to be used for reservations. It can be a value from 0 through 65000 percent. If you specify a value greater than 100, you are oversubscribing the interface.

You can use the subscription factor to shut down new RSVP sessions on a per-interface basis. If you set the percentage to 0, no new sessions (including those with zero bandwidth requirements) are permitted on the interface. Existing RSVP sessions are not affected by changing the subscription factor. To clear an existing session, issue the clear rsvp session command.

## Configure the RSVP Update Threshold on an Interface

The interior gateway protocols (IGPs) maintain the traffic engineering database (TED), but the current available bandwidth on the TED links originates from RSVP. When a link's bandwidth changes, RSVP informs the IGPs, which can then update the TED and forward the new bandwidth information to all network nodes. The network nodes then know how much bandwidth is available on the TED link (local or remote), and CSPF can correctly compute the paths.

However, IGP updates can consume excessive system resources. Depending on the number of nodes in a network, it might not be desirable to perform an IGP update for small changes in bandwidth. By configuring the update-threshold statement at the [edit protocols rsvp] hierarchy level, you can adjust the threshold at which a change in bandwidth triggers an IGP update.

You can configure a value of between 1 percent and 20 percent (the default is 10 percent) for when to trigger an IGP update. For example, if you have configured the update-threshold statement to be 15 percent and the router discovers that the bandwidth on a link has changed by 10 percent, RSVP does not trigger an IGP update. However, if the bandwidth on a link changes by 20 percent, RSVP does trigger an IGP update.

Configure the update-threshold statement at the [edit protocols rsvp] hierarchy level:

```
interface interface-name {
  update-threshold threshold;
}
```

Because of the update threshold, it is possible for Constrain Shortest Path First (CSPF) to compute a path using outdated TED bandwidth information on a link. If RSVP attempts to establish an LSP over that path, it might find that there is insufficient bandwidth on that link. When this happens, RSVP triggers an IGP TED update, flooding the updated bandwidth information on the network. CSPF can then recompute the path using the updated bandwidth information, and can then attempt to find a different path, avoiding the congested link. Note that this functionality is the default and does not need any additional configuration.

You can configure the rsvp-error-hold-time statement at the [edit protocols mpls] hierarchy level to improve the accuracy of the TED database (including the accuracy of bandwidth estimates for LSPs) using information provided by PathErr messages. See “Improving TED Accuracy with RSVP PathErr Messages” on page 75.

## Configure RSVP Timers

RSVP uses two interrelated timing parameters:

The refresh time controls the interval between the successive generation of refresh messages. Refresh messages include Path and Resv messages. Refresh messages are sent periodically so that reservation states in neighboring nodes do not time out. Each node chooses a value for the refresh timer independently. Each Path and Resv message carries the refresh timer value, and the receiving node extracts this value from the messages.

The keep multiplier is a locally configured small integer in the range 1 through 255.

To determine the lifetime of a reservation state, use the following formula:

$$\textit{lifetime} = (\textit{keep-multiplier} + 0.5) * 1.5 * \textit{refresh-time}$$

In the worst case,  $(\textit{keep-multiplier} - 1)$  successive refresh messages must be lost before a reservation state is deleted.

By default, the refresh timer value is 30 seconds. To modify this value, include the refresh-time statement at the [edit protocols rsvp] hierarchy level:

```
[edit protocols rsvp]
refresh-time seconds;
```

The default value of the keep multiplier is 3. To modify this value, include the keep-multiplier statement at the [edit protocols rsvp] hierarchy level:

```
[edit protocols rsvp]
keep-multiplier number;
```

## Preempt RSVP Sessions

Whenever bandwidth is insufficient to handle all RSVP sessions, you can control the preemption of RSVP sessions. By default, an RSVP session is preempted only by a new higher-priority session.

To always preempt a session when the bandwidth is insufficient, include the preemption aggressive statement at the [edit protocols rsvp] hierarchy level:

```
[edit protocols rsvp]
preemption aggressive;
```

To disable RSVP session preemption, include the preemption disabled statement at the [edit protocols rsvp] hierarchy level:

```
[edit protocols rsvp]
preemption disabled;
```

To return to the default (that is, preempt a session only for a new higher-priority session), include the preemption normal statement at the [edit protocols rsvp] hierarchy level:

```
[edit protocols rsvp]
preemption normal;
```

## Trace RSVP Protocol Traffic

To trace RSVP protocol traffic, you can specify options in the global traceoptions statement at the [edit routing-options] hierarchy level, and you can specify RSVP-specific options by including the traceoptions statement at the [edit protocols rsvp] hierarchy level:

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

Use the file statement to specify the name of the file that receives the output of the tracing operation. All files are placed in the directory /var/log. We recommend that you place RSVP tracing output in the file rsvp-log.

You can specify the following RSVP-specific flags in the RSVP traceoptions statement:

all—All tracing operations.

error—Trace all detected error conditions.

packets—Trace all RSVP messages, including Path, Resv, PathTear, ResvTear, PathErr, ResvErr, and ResvConf messages.

path—Trace Path messages.

pathtear—Trace PathTear messages.

resv—Trace Resv messages.

resvtear—Trace ResvTear messages.

route—Trace routing table changes in RSVP.

state—Trace session state transitions.

For general information about tracing and global tracing options, see the *JUNOS Internet Software Configuration Guide: Routing and Routing Protocols*.

## Examples: Trace RSVP Protocol Traffic

Trace RSVP Path messages in detail:

```
[edit]
protocols {
  rsvp {
    traceoptions {
      file rsvp size 10m files 5;
      flag path;
    }
  }
}
```

Trace all RSVP messages:

```
[edit]
protocols {
  rsvp {
    traceoptions {
      file rsvp size 10m files 5;
      flag packets;
    }
  }
}
```

Trace all RSVP error conditions:

```
[edit]
protocols {
  rsvp {
    traceoptions {
      file rsvp size 10m files 5;
      flag error;
    }
  }
}
```

## Configure RSVP and MPLS

The primary purpose of the JUNOS RSVP software is to support dynamic signaling within LSPs. When you enable both MPLS and RSVP on a router, MPLS becomes a client of RSVP. No additional configuration is required to bind MPLS and RSVP.

You can configure MPLS to set up signaled paths using the label-switched-path statement at the [edit protocols mpls] hierarchy level. Each LSP translates into a request for RSVP to initiate an RSVP session. This request is passed through the internal interface between label switching and RSVP. After examining the request information, checking RSVP states and the local routing tables, RSVP initiates one session for each LSP. The session is sourced from the local router and is destined to the target of the LSP.

When an RSVP session is successfully created, the LSP is set up along the paths created by the RSVP session. If the RSVP session is unsuccessful, RSVP notifies MPLS of its status. It is up to MPLS to try to initiate backup paths or to continue retrying the initial path.

To pass label-switching signaling information, RSVP supports four additional objects: Label Request Object, Label Object, Explicit Route Object, and Record Route Object. For an LSP to be set up successfully, all routers along the path must support MPLS, RSVP, and these four objects. Of the four objects, Record Route Object is not mandatory.

To configure MPLS and make it a client of RSVP, do the following:

Enable MPLS on all routers that are to participate in label switching (that is, on all routers that might be part of a label-switching path).

Enable RSVP on all routers and on all router interfaces that form the LSP.

Configure the routers that are to be the beginning of the LSP.

### **Example: Configure RSVP and MPLS**

The following shows a sample configuration for a router at the beginning of an LSP:

```
[edit]
protocols {
  mpls {
    label-switched-path sf-to-london {
      to 192.168.1.4;
    }
  }
  rsvp {
    interface so-0/0/0;
  }
}
```

The following shows a sample configuration for all the other routers that form the LSP:

```
[edit]
protocols {
  mpls {
    interface so-0/0/0;
  }
  rsvp {
    interface so-0/0/0;
  }
}
```

