



# Implementing RSVP for MPLS-TE

- [Implementing RSVP for MPLS-TE , on page 1](#)

## Implementing RSVP for MPLS-TE

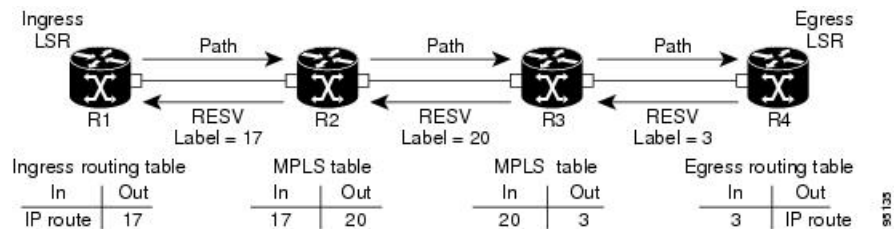
Resource Reservation Protocol (RSVP) is a signaling protocol that enables systems to request resource reservations from the network. RSVP processes protocol messages from other systems, processes resource requests from local clients, and generates protocol messages. As a result, resources are reserved for data flows on behalf of local and remote clients. RSVP creates, maintains, and deletes these resource reservations.

MPLS Traffic Engineering (MPLS-TE) learns the topology and resources available in a network and then maps traffic flows to particular paths based on resource requirements and network resources such as bandwidth. MPLS TE builds a unidirectional tunnel from a source to a destination in the form of a label switched path (LSP), which is then used to forward traffic. MPLS-TE uses RSVP to signal LSPs.

## Setting up MPLS LSP Using RSVP

The following figure shows how RSVP sets up a LSP from router R1 through router R4 that can be used for TE in an MPLS environment.

**Figure 1: MPLS LSP Using RSVP**



The LSP setup is initiated when the LSP head node sends path messages to the tail node. The Path messages reserve resources along the path to each node, and creates path states associated with the session on each node. When the tail node receives a path message, it sends a reservation (RESV) message with a label back to the previous node. The reservation state in each router is considered as a soft state, which means that periodic PATH and RESV messages must be sent at each hop to maintain the state.

When the reservation message arrives at the previous node, it causes the reserved resources to be locked and forwarding entries are programmed with the MPLS label sent from the tail-end node. A new MPLS label is

allocated and sent to the next node upstream. When the reservation message reaches the head node, the label is programmed and the MPLS data starts to flow along the path.

## Overview of RSVP for MPLS-TE Features

This section provides an overview of the various features of RSVP for MPLS-TE.

RSVP is automatically enabled on interfaces on which MPLS-TE is configured. For MPLS-TE LSPs with bandwidth, the RSVP bandwidth has to be configured on the interfaces. There is no need to configure RSVP, if all MPLS-TE LSPs have zero bandwidth.

RSVP Graceful restart ensures high availability and allows RSVP TE enabled routers to recover RSVP state information from neighbors after a failure in the network.

RSVP requires that the path and reservation state that are set up during LSP signaling must be refreshed by periodically sending refresh messages. Refresh messages are used to synchronize the state between RSVP neighbors and to recover from lost RSVP messages. RSVP refresh reduction feature includes support for reliable messages which are transmitted rapidly when the messages are lost. Summary refresh messages contain information to refresh multiple states and reduces the number of messages required to refresh states.

RSVP messages can be authenticated to ensure that only trusted neighbors can set up reservations.

For detailed information about RSVP for MPLS-TE features, see *RSVP for MPLS-TE Features- Details*.

## Configuring RSVP for MPLS-TE

RSVP requires coordination among several routers, establishing exchange of RSVP messages to set up LSPs. To configure RSVP, you need to install the following two RPMs :

- ncs540-mpls-2.0.0.0-r601.x86\_64.rpm-6.0.1
- ncs540-mpls-te-rsvp-2.0.0.0-r601.x86\_64.rpm-6.0.1

Depending on the requirements, RSVP requires some basic configuration described in the following topics:

### Configuring RSVP Message Authentication Globally

The RSVP authentication feature permits neighbors in an RSVP network to use a secure hash algorithm to authenticate all RSVP signaling messages digitally. The authentication is accomplished on a per-RSVP-hop basis using an RSVP integrity object in the RSVP message. The integrity object includes a key ID, a sequence number for messages, and keyed message digest.

You can globally configure the values of authentication parameters including the key-chain, time interval that RSVP maintains security associations with other trusted RSVP neighbors (life time) and maximum number of RSVP authenticated messages that can be received out of sequence (window size). These defaults are inherited for each neighbor or interface.

#### Configuration Example

In this example, authentication parameters are configured globally on a router. The authentication parameters including authentication key-chain, lifetime, and window size are configured. A valid key-chain should be configured before performing this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# key chain mpls-keys
```

```
RP/0/RP0/CPU0:router(config-mpls-keys)# exit
RP/0/RP0/CPU0:router(config)# rsvp authentication
RP/0/RP0/CPU0:router(config-rsvp-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-auth)# window-size 33
```

### Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail
```

```
RSVP Authentication Information:
  Source Address:      3.0.0.1
  Destination Address: 3.0.0.2
  Neighbour Address:   3.0.0.2
  Interface:           HundredGigabitEthernet 0/0/0/3
  Direction:           Send
  LifeTime:             2000 (sec)
  LifeTime left:        1305 (sec)
  KeyType:              Static Global KeyChain
  Key Source:           mpls-keys
  Key Status:           No error
  KeyID:                1
  Digest:               HMAC MD5 (16)
  window-size:         33
Challenge:              Not supported
TX Sequence:           5023969459702858020 (0x45b8b99b00000124)
Messages successfully authenticated: 245
Messages failed authentication: 0
```

### Related Topics

- [Configuring RSVP Authentication for an Interface, on page 3](#)
- [Configuring RSVP Authentication on a Neighbor, on page 4](#)
- [#unique\\_60](#)

## Configuring RSVP Authentication for an Interface

You can individually configure the values of RSVP authentication parameters including key-chain, life time, and window size on an interface. Interface specific authentication parameters are used to secure specific interfaces between two RSVP neighbors.

### Configuration Example

This example configures authentication key-chain, life time for the security association, and window size on an interface. A valid key-chain should be already configured to use it as part of this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# authentication
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# window-size 33
RP/0/RP0/CPU0:router(config)# commit
```

### Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail

RSVP Authentication Information:
  Source Address:      3.0.0.1
  Destination Address: 3.0.0.2
  Neighbour Address:   3.0.0.2
  Interface:           HundredGigabitEthernet 0/0/0/3
  Direction:           Send
  LifeTime:             2000 (sec)
  LifeTime left:       1305 (sec)
  KeyType:              Static Global KeyChain
  Key Source:           mpls-keys
  Key Status:           No error
  KeyID:                1
  Digest:               HMAC MD5 (16)
  window-size:         33
  Challenge:            Not supported
  TX Sequence:         5023969459702858020 (0x45b8b99b00000124)
  Messages successfully authenticated: 245
  Messages failed authentication: 0
```

### Related Topics

- [Configuring RSVP Message Authentication Globally, on page 2](#)
- [Configuring RSVP Authentication on a Neighbor, on page 4](#)
- [#unique\\_60](#)

## Configuring RSVP Authentication on a Neighbor

You can individually configure the values of RSVP authentication parameters including key-chain, life time, and window size on a neighbor.

### Configuration Example

This example configures the authentication key-chain, life time for the security association, and window size on a RSVP neighbor. A valid key-chain should be already configured to use it as part of this task.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp neighbor 1.1.1.1 authentication
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# key-source key-chain mpls-keys
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# life-time 2000
RP/0/RP0/CPU0:router(config-rsvp-if-auth)# window-size 33
RP/0/RP0/CPU0:router(config)# commit
```

### Verification

Verify the configuration of authentication parameters using the following command.

```
RP/0/RP0/CPU0:router# show rsvp authentication detail

RSVP Authentication Information:
  Neighbour Address:   1.1.1.1
  Interface:           HundredGigabitEthernet 0/0/0/3
  Direction:           Send
  LifeTime:             2000 (sec)
  LifeTime left:       1205 (sec)
  KeyType:              Static Global KeyChain
  Key Source:           mpls-keys
  Key Status:           No error
  KeyID:                1
  Digest:               HMAC MD5 (16)
```

```

window-size:          33
Challenge:            Not supported

```

### Related Topics

- [Configuring RSVP Message Authentication Globally, on page 2](#)
- [Configuring RSVP Authentication for an Interface, on page 3](#)
- [#unique\\_60](#)

## Configuring Graceful Restart

RSVP graceful restart provides a mechanism to ensure high availability (HA), which allows detection and recovery from failure conditions for systems running Cisco IOS XR software and ensures non-stop forwarding services. RSVP graceful restart is based on RSVP hello messages and allows RSVP TE enabled routers to recover RSVP state information from neighbors after a failure in the network. RSVP uses a Restart Cap object (RSVP RESTART) in hello messages in which restart and recovery times are specified to advertise the restart capability of a node. The neighboring node helps a restarting node by sending a Recover Label object to recover the forwarding state of the restarting node.

You can configure standard graceful restart which is based on node-id address based hello messages and also interface-based graceful restart which is interface-address based hello messages.

### Configuration Example

In this example, RSVP-TE is already enabled on the router nodes on a network and graceful restart needs to be enabled on the router nodes for failure recovery. Graceful restart is configured globally to enabled node-id address based hello messages and also on a router interface to support interface-address based hello messages.

```

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling graceful-restart
RP/0/RP0/CPU0:router(config-rsvp)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling graceful-restart
interface-based
RP/0/RP0/CPU0:router(config)# commit

```

### Verification

Use the following commands to verify that graceful restart is enabled.

```

RP/0/RP0/CPU0:router# show rsvp graceful-restart
Graceful restart: enabled Number of global neighbors: 1
Local MPLS router id: 192.168.55.55
Restart time: 60 seconds Recovery time: 120 seconds
Recovery timer: Not running
Hello interval: 5000 milliseconds Maximum Hello miss-count: 4

RP/0/RP0/CPU0:router# show rsvp graceful-restart neighbors detail

Neighbor: 192.168.77.77 Source: 192.168.55.55 (MPLS)
Hello instance for application MPLS
Hello State: UP (for 00:20:52)
Number of times communications with neighbor lost: 0
Reason: N/A
Recovery State: DONE

```

```

Number of Interface neighbors: 1
address: 192.168.55.0
Restart time: 120 seconds Recovery time: 120 seconds
Restart timer: Not running
Recovery timer: Not running
Hello interval: 5000 milliseconds Maximum allowed missed Hello messages: 4

```

### Related Topics

- [#unique\\_60](#)

## Configuring Refresh Reduction

RSVP Refresh Reduction improves the reliability of Resource Reservation Protocol (RSVP) signaling to enhance network performance and message delivery and it is enabled by default. Refresh reduction is used with a neighbor only if the neighbor supports it. You can also disable refresh reduction on an interface if you want.

### Configuration Example

The example shows how to configure the various parameters available for the refresh reduction feature.

The following parameters are configured to change their default values:

- refresh interval
- number of refresh messages a node can miss
- retransmit time
- acknowledgment hold time
- acknowledgment message size
- refresh message summary size

```

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh interval 40
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh missed 6
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh reduction reliable retransmit-time
2000
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh reduction reliable ack-hold-time
1000
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh reduction reliable ack-max-size
1000
RP/0/RP0/CPU0:router(config-rsvp-if)# signalling refresh reduction summary max-size 1500
RP/0/RP0/CPU0:router(config)# commit

```

## Configuring ACL Based Prefix Filtering

You can configure extended access lists (ACLs) to forward, drop, or perform normal processing on RSVP router-alert (RA) packets. For each incoming RSVP RA packet, RSVP inspects the IP header and attempts to match the source or destination IP addresses with a prefix configured in an extended ACL. If there is no explicit permit or explicit deny, the ACL infrastructure returns an implicit deny by default. By default, RSVP processes the packet if the ACL match yields an implicit (default) deny.

### Configuration Example

This example configures ACL based prefix filtering on RSVP RA packets. When RSVP receives a RA packet from source address 1.1.1.1 it is forwarded and packets destined to the IP address 2.2.2.2 are dropped.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list rsvpac1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 host 1.1.1.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ipv4 any host 2.2.2.2
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering access-list rsvp-acl
RP/0/RP0/CPU0:router(config)# commit
```

### Verification

Verify the configuration of ACL based prefix filtering

```
RP/0/RP0/CPU0:router# show rsvp counters prefix-filtering access-list rsvp-acl
```

ACL:rsvp-acl	Forward	Local	Drop	Total
Path	0	0	0	0
PathTear	0	0	0	0
ResvConfirm	0	0	0	0
Total	0	0	0	0

### Related Topics

- [Configuring RSVP Packet Dropping, on page 7](#)

## Configuring RSVP Packet Dropping

You can configure extended access lists (ACLs) to forward, drop, or perform normal processing on RSVP router-alert (RA) packets. By default, RSVP processes the RA packets even if the ACL match yields an implicit deny. You can configure RSVP to drop RA packets when the ACL matches results in an implicit deny.

### Configuration Example

This example configures ACL based prefix filtering on RSVP RA packets. When RSVP receives a RA packet from source address 1.1.1.1 it is forwarded and packets destined to the IP address 2.2.2.2 are dropped. RA packets are dropped if the ACL matches results in an implicit deny.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list rsvpac1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 host 1.1.1.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ipv4 any host 2.2.2.2
RP/0/RP0/CPU0:router(config)# rsvp
RP/0/RP0/CPU0:router(config-rsvp)# signalling prefix-filtering default-deny-action drop
RP/0/RP0/CPU0:router(config)# commit
```

### Verification

Verify the configuration of RSVP packet drop using the following command.

```
RP/0/RP0/CPU0:router# show rsvp counters prefix-filtering access-list rsvpac1
```

ACL: rsvpac1	Forward	Local	Drop	Total
Path	4	1	0	5
PathTear	0	0	0	0

ResvConfirm	0	0	0	0
Total	4	1	0	5

### Related Documents

- [Configuring ACL Based Prefix Filtering, on page 6](#)

## Enabling RSVP Traps

By implementing the RSVP MIB, you can use SNMP to access objects belonging to RSVP. You can also specify two traps (NewFlow and LostFlow) which are triggered when a new flow is created or deleted. RSVP MIBs are automatically enabled when you turn on RSVP, but you need to enable RSVP traps.

### Configuration Example

This example shows how to enable RSVP MIB traps when a flow is deleted or created and also how to enable both the traps.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp lost-flow
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp new-flow
RP/0/RP0/CPU0:router(config)# snmp-server traps rsvp all
RP/0/RP0/CPU0:router(config)# commit
```

## RSVP for MPLS-TE Features- Details

### RSVP Graceful Restart Operation

RSVP graceful restart is based on RSVP hello messages. Hello messages are exchanged between the router and its neighbor nodes. Each neighbor node can autonomously issue a hello message containing a hello request object. A receiver that supports the hello extension replies with a hello message containing a hello acknowledgment (ACK) object. If the sending node supports state recovery, a Restart Cap object that indicates a node's restart capability is also carried in the hello messages. In the Restart Cap object, the restart time and the recovery time is specified. The restart time is the time after a loss in Hello messages within which RSVP hello session can be re-established. The recovery time is the time that the sender waits for the recipient to re-synchronize states after the re-establishment of hello messages.

For graceful restart, the hello messages are sent with an IP Time to Live (TTL) of 64. This is because the destination of the hello messages can be multiple hops away. If graceful restart is enabled, hello messages (containing the restart cap object) are sent to an RSVP neighbor when RSVP states are shared with that neighbor. If restart cap objects are sent to an RSVP neighbor and the neighbor replies with hello messages containing the restart cap object, the neighbor is considered to be graceful restart capable. If the neighbor does not reply with hello messages or replies with hello messages that do not contain the restart cap object, RSVP backs off sending hellos to that neighbor. If a hello Request message is received from an unknown neighbor, no hello ACK is sent back.

### RSVP Authentication

Network administrators need the ability to establish a security domain to control the set of systems that initiates RSVP requests. The RSVP authentication feature permits neighbors in an RSVP network to use a secure hash to sign all RSVP signaling messages digitally, thus allowing the receiver of an RSVP message to verify the sender of the message without relying solely on the sender's IP address.



The signature is accomplished on a per-RSVP-hop basis with an RSVP integrity object in the RSVP message as defined in RFC 2747. The integrity object includes a key ID, a sequence number for messages, and keyed message digest. This method provides protection against forgery or message modification. However, the receiver must know the security key used by the sender to validate the digital signature in the received RSVP message. Network administrators manually configure a common key for each RSVP neighbor on the shared network. The sending and receiving systems maintain a security association for each authentication key that they share. For detailed information about different security association parameters, see the **Security Association Parameters** table.

You can configure global defaults for all authentication parameters including key, window size, and lifetime. These defaults are inherited when you configure authentication for each neighbor or interface. However, you can also configure these parameters individually on a neighbor or interface basis, in which case the global values (configured or default) are no longer inherited.

Interface and neighbor interface modes unless explicitly configured, inherit the parameters from global configuration mode as follows:

- Window-size is set to 1.
- Lifetime is set to 1800.
- key-source key-chain command is set to none or disabled.

The following situations explain how to choose between global, interface, or neighbor configuration modes:

- Global configuration mode is optimal when a router belongs to a single security domain (for example, part of a set of provider core routers). A single common key set is expected to be used to authenticate all RSVP messages.
- Interface, or neighbor configuration mode, is optimal when a router belongs to more than one security domain. For example, a provider router is adjacent to the provider edge (PE), or a PE is adjacent to an edge device. Different keys can be used but not shared.

A security association (SA) is a collection of information that is required to maintain secure communications with a peer. The following table lists the main parameters that defines a security association

**Table 1: Security Association Parameters**

Security Association Parameters	Description
src	IP address of the sender.
dst	IP address of the final destination.
interface	Interface of the security association.
direction	Send or receive type of the security association.
Lifetime	Expiration timer value that is used to collect unused security association data.
Sequence Number	Last sequence number that was either sent or accepted (dependent of the direction type).
key-source	Source of keys for the configurable parameter.
keyID	Key number (returned from the key-source) that was last used.

Security Association Parameters	Description
Window Size	Specifies the maximum number of authenticated messages that can be received out of order.
Window	Specifies the last <i>window size</i> value sequence number that is received or accepted.