CONTENTS

Preface vii
Changes to This Document vii

CHAPTER 1
Implementing MPLS Label Distribution Protocol 1
Implementing MPLS Label Distribution Protocol 1
Prerequisites for Implementing MPLS Label Distribution Protocol 1
Overview of Label Distribution Protocol 2
Configuring Label Distribution Protocol 2
  Configuring Label Distribution Protocol 2
  Configuring Label Distribution Protocol Discovery Parameters 3
Label Distribution Protocol Discovery for Targeted Hellos 3
Label Advertisement Control 4
Configuring Local Label Allocation Control 5
Configuring Downstream on Demand 5
Configuring Explicit Null Label 6
Label Distribution Protocol Auto-configuration 6
Configuring Session Protection 7
Configuring Label Distribution Protocol- Interior Gateway Protocol (IGP) Synchronization 7
Configuring Label Distribution Protocol Graceful Restart 8
Configuring Label Distribution Protocol Nonstop Routing 8
MPLS Label Distribution Protocol : Details 9
  Setting Up Label Switched Paths 9
  Details of Label Distribution Protocol Graceful Restart 12
  Details of Session Protection 15

CHAPTER 2
Implementing MPLS Static Labeling 17
MPLS Static Labeling 17
CHAPTER 3
Implementing RSVP for MPLS-TE 21
Implementing RSVP for MPLS-TE 21
Setting up MPLS LSP Using RSVP 21
Overview of RSVP for MPLS-TE Features 22
Configuring RSVP for MPLS-TE 22
  Configuring RSVP Message Authentication Globally 22
  Configuring RSVP Authentication for an Interface 23
  Configuring RSVP Authentication on a Neighbor 24
  Configuring Graceful Restart 25
  Configuring Refresh Reduction 26
  Configuring ACL Based Prefix Filtering 26
  Configuring RSVP Packet Dropping 27
Enabling RSVP Traps 28
RSVP for MPLS-TE Features- Details 28

CHAPTER 4
Implementing MPLS Traffic Engineering 31
Implementing MPLS Traffic Engineering 31
Overview of MPLS-TE Features 32
How MPLS-TE Works 32
Configuring MPLS-TE 33
  Building MPLS-TE Topology 33
  Creating an MPLS-TE Tunnel 34
  Configuring Fast Reroute 35
  Configuring AutoTunnel Backup 36
  Configuring Next Hop Backup Tunnel 37
  Configuring SRLG Node Protection 37
  Configuring Pre-Standard DS-TE 38
  Configuring an IETF DS-TE Tunnel Using RDM 39
  Configuring an IETF DS-TE Tunnel Using MAM 40
  Configuring Flexible Name-Based Tunnel Constraints 40
  Configuring Automatic Bandwidth 41
  Configuring AutoTunnel Mesh 42
Configuring an MPLS Traffic Engineering Interarea Tunneling  43

MPLS-TE Features - Details  44

CHAPTER 5

Implementing MPLS OAM  49
Implementing MPLS OAM  49
MPLS LSP Ping  49
MPLS LSP Traceroute  51
Preface

• Changes to This Document, page vii

Changes to This Document

The following table lists the technical changes made to this document since it was first published.

<table>
<thead>
<tr>
<th>Date</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>November 2016</td>
<td>Initial release of this document.</td>
</tr>
</tbody>
</table>
Implementing MPLS Label Distribution Protocol

MPLS (Multi Protocol Label Switching) is a forwarding mechanism based on label switching. In an MPLS network, data packets are assigned labels and packet-forwarding decisions are taken based on the contents of the label. To switch labeled packets across the MPLS network, predetermined paths are established for various source-destination pairs. These predetermined paths are known as Label Switched Paths (LSPs). To establish LSPs, MPLS signaling protocols are used. Label Distribution Protocol (LDP) is an MPLS signaling protocol used for establishing LSPs. This module provides information about how to configure MPLS LDP.

Prerequisites for Implementing MPLS Label Distribution Protocol

The following are the prerequisites to implement MPLS LDP:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must be running Cisco IOS XR software.
- You must install a composite mini-image and the MPLS package.
- You must activate IGP.
We recommend to use a lower session holdtime bandwidth such as neighbors so that a session down occurs before an adjacency-down on a neighbor. Therefore, the following default values for the hello times are listed:

- Holdtime is 15 seconds.
- Interval is 5 seconds.

For example, the LDP session holdtime can be configured as 30 seconds by using the `holdtime` command.

### Overview of Label Distribution Protocol

In IP forwarding, when a packet arrives at a router the router looks at the destination address in the IP header, performs a route lookup, and then forwards the packet to the next hop. MPLS is a forwarding mechanism in which packets are forwarded based on labels. Label Distribution Protocols assign, distribute, and install the labels in an MPLS environment. It is the set of procedures and messages by which Label Switched Routers (LSRs) establish LSPs through a network by mapping network-layer routing information directly to data-link layer switched paths. These LSPs may have an endpoint at a directly attached neighbor (comparable to IP hop-by-hop forwarding), or may have an endpoint at a network egress node, enabling switching via all intermediary nodes.

LSPs can be created statically, by RSVP traffic engineering (TE), or by LDP. LSPs created by LDP perform hop-by-hop path setup instead of an end-to-end path. LDP enables LSRs to discover their potential peer routers and to establish LDP sessions with those peers to exchange label binding information. Once label bindings are learned, the LDP is ready to setup the MPLS forwarding plane.

For more information about setting up LSPs, see [MPLS Label Distribution Protocol: Details](#), on page 9.

### Configuring Label Distribution Protocol

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

#### Configuration Example

This example shows how to enable LDP over an interface.

```
RP/0/RP0/CPU0:Router(config)#  mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)#  router-id 192.168.70.1
RP/0/RP0/CPU0:Router(config-ldp)#  interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ldp-if)#  commit
```
Configuring Label Distribution Protocol Discovery Parameters

LSRs that are running LDP sends hello messages on all the LDP enabled interfaces to discover each other. So, the LSR that receives the LDP hello message on an interface is aware of the presence of the LDP router on that interface. If LDP hello messages are sent and received on an interface, there is an LDP adjacency across the link between the two LSRs that are running LDP. By default, hello messages are sent every 5 seconds with a hold time of 15 seconds. If the LSR does not receive a discovery hello from peer before the hold time expires, the LSR removes the peer LSR from the list of discovered LDP neighbors. The LDP discovery parameters can be configured to change the default parameters.

LDP session between LSRs that are not directly connected is known as targeted LDP session. For targeted LDP sessions, LDP uses targeted hello messages to discover the extended neighbors. By default, targeted hello messages are sent every 10 seconds with a hold time of 90 seconds.

**Configuration Example**

This example shows how to configure the following LDP discovery parameters:

- hello hold time
- hello interval
- targeted hello hold time
- targeted hello interval

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# router-id 192.168.70.1
RP/0/RP0/CPU0:Router(config-ldp)# discovery hello holdtime 30
RP/0/RP0/CPU0:Router(config-ldp)# discovery hello interval 10
RP/0/RP0/CPU0:Router(config-ldp)# discovery targeted-hello holdtime 120
RP/0/RP0/CPU0:Router(config-ldp)# discovery targeted-hello interval 15
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

**Verification**

This section verifies the MPLS LDP discovery parameters configuration.

```
RP/0/RP0/CPU0:Router# show mpls ldp parameters
LD Parameters:
Role: Active
Protocol Version: 1
Router ID: 192.168.70.1
Discovery:
Link Hellos: Holdtime:30 sec, Interval:10 sec
Targeted Hellos: Holdtime:120 sec, Interval:15 sec
Quick-start: Enabled (by default)
Transport address: IPv4: 192.168.70.1
```

**Label Distribution Protocol Discovery for Targeted Hellos**

LDP session between LSRs that are not directly connected is known as targeted LDP session. For LDP neighbors which are not directly connected, you should manually configure the LDP neighborship on both the routers.
Label Advertisement Control

LDP allows you to control the advertising and receiving of labels. You can control the exchange of label binding information by using label advertisement control (outbound filtering) or label acceptance control (inbound filtering).

Label Advertisement Control (Outbound Filtering)

Label Distribution Protocol advertises labels for all the prefixes to all its neighbors. When this is not desirable (for scalability and security reasons), you can configure LDP to perform outbound filtering for local label advertisement for one or more prefixes to one or more peers. This feature is known as LDP outbound label filtering, or local label advertisement control. You can control the exchange of label binding information using the `mpls ldp label advertise` command. Using the optional keywords, you can advertise selective prefixes to all neighbors, advertise selective prefixes to defined neighbors, or disable label advertisement to all peers for all prefixes. Prefixes and peers advertised selectively are defined in the access list.

Configuration Example: Label Advertisement Control

This example shows how to configure outbound label advertisement control. In this example, neighbors are specified to advertise and receive label advertisements. Also an interface is specified for label advertisement.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise to 1.1.1.1:0 for pfx_acl1
```

Label Acceptance Control (Inbound Filtering)

LDP accepts labels (as remote bindings) for all prefixes from all peers. LDP operates in liberal label retention mode, which instructs LDP to keep remote bindings from all peers for a given prefix. For security reasons, or to conserve memory, you can override this behavior by configuring label binding acceptance for set of prefixes from a given peer. The ability to filter remote bindings for a defined set of prefixes is also referred to as LDP inbound label filtering or label acceptance control.
Configuration Example : Label Acceptance Control (Inbound Filtering)

This example shows how to configure label acceptance control. In this example, an LSR is configured to accept and retain label bindings from neighbors for prefixes defined in access list.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label remote accept from 192.168.1.1:0 for acl_1
RP/0/RP0/CPU0:Router(config-ldp-af)# label remote accept from 192.168.2.2:0 for acl_2
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Configuring Local Label Allocation Control

LDP creates label bindings for all IGP prefixes and receives label bindings for all IGP prefixes from all its peers. If an LSR receives label bindings from several peers for thousands of IGP prefixes, it consumes significant memory and CPU. In some scenarios, most of the LDP label bindings may not be useful for any application and you may require to limit the allocation of local labels. This is accomplished using LDP local label allocation control, where an access list can be used to limit allocation of local labels to a set of prefixes. Limiting local label allocation provides several benefits, including reduced memory usage requirements, fewer local forwarding updates, and fewer network and peer updates.

**Configuration Example**

This example shows how to configure local label allocation using an IP access list to specify a set of prefixes that local labels can allocate and advertise.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local allocate for pfx_acl_1
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Configuring Downstream on Demand

By default, LDP uses downstream unsolicited mode in which label advertisements for all routes are received from all LDP peers. The downstream on demand feature adds support for downstream-on-demand mode, where the label is not advertised to a peer, unless the peer explicitly requests it. At the same time, since the peer does not automatically advertise labels, the label request is sent whenever the next-hop points out to a peer that no remote label has been assigned.

In downstream on demand configuration, an ACL is used to specify the set of peers for downstream on demand mode. For downstream on demand to be enabled, it needs to be configured on both peers of the session. If only one peer in the session has downstream-on-demand feature configured, then the session does not use downstream-on-demand mode.

**Configuration Example**

This example shows how to configure LDP Downstream on Demand.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label remote accept from 192.168.1.1:0 for acl_1
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```
Configuring Explicit Null Label

Cisco MPLS LDP uses implicit or explicit null label as local label for routes or prefixes that terminate on the given LSR. These routes include all local, connected, and attached networks. By default, the null label is implicit-null that allows LDP control plane to implement penultimate hop popping (PHP) mechanism. When this is not desirable, you can configure explicit-null label that allows LDP control plane to implement ultimate hop popping (UHP) mechanism. You can configure explicit-null feature on the ultimate hop LSR. Access-lists can be used to specify the IP prefixes for which PHP is desired.

You can enforce implicit-null local label for a specific prefix by using the implicit-null-override command even if the prefix requires a non-null label to be allocated by default. For example, by default, an LSR allocates and advertises a non-null label for an IGP route. If you wish to terminate LSP for this route on penultimate hop of the LSR, you can enforce implicit-null label allocation and advertisement for this prefix using the implicit-null-override command.

Configuration Example: Explicit Null

This example shows how to configure explicit null label.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise explicit-null
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Configuration Example: Implicit Null Override

This example shows how to configure implicit null override for a set of prefixes.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# address-family ipv4
RP/0/RP0/CPU0:Router(config-ldp-af)# label local advertise implicit-null-override for acl-1
RP/0/RP0/CPU0:Router(config-ldp-af)# commit
```

Label Distribution Protocol Auto-configuration

LDP auto-configuration allows you to automatically configure LDP on all interfaces for which the IGP protocol is enabled. Typically, LDP assigns and advertises labels for IGP routes and must often be enabled on all active interfaces by an IGP. During LDP manual configuration, you must define the set of interfaces under LDP which is a time-intensive procedure. LDP auto-configuration eliminates the need to specify the same list of interfaces under LDP and simplifies the configuration tasks.

Configuration Example: Enabling LDP Auto-Configuration for OSPF

This example shows how to enable LDP auto-configuration for a specified OSPF instance.

```
RP/0/RP0/CPU0:Router(config)# router ospf 190
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp auto-config
RP/0/RP0/CPU0:Router(config-ospf)# area 8
RP/0/RP0/CPU0:Router(config-ospf-area)# interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ospf-area-if)# commit
```
Configuring Session Protection

When a new link or node comes up after a link failure, IP converges earlier and much faster than MPLS LDP and may result in MPLS traffic loss until the MPLS convergence. If a link flaps, the LDP session also flaps due to loss of link discovery. LDP session protection minimizes traffic loss, provides faster convergence, and protects existing LDP (link) sessions. When session protection is enabled for a peer, LDP starts sending targeted hello (directed discovery) in addition to basic discovery link hellos. When the direct link goes down, the targeted hellos can still be forwarded to the peer LSR over an alternative path as long as there is one. So, the LDP session stays up after the link goes down.

You can configure LDP session protection to automatically protect sessions with all or a given set of peers (as specified by peer-acl). When configured, LDP initiates backup targeted hellos automatically for neighbors for which primary link adjacencies already exist. These backup targeted hellos maintain LDP sessions when primary link adjacencies go down.

Configuration Example

This example shows how to configure LDP session protection for peers specified by the access control list peer-acl-1 for a maximum duration of 60 seconds.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# session protection for peer-acl-1 duration 60
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Configuring Label Distribution Protocol- Interior Gateway Protocol (IGP) Synchronization

Lack of synchronization between LDP and Interior Gateway Protocol (IGP) can cause MPLS traffic loss. Upon link up, for example, IGP can advertise and use a link before LDP convergence has occurred or, a link may continue to be used in IGP after an LDP session goes down.

LDP IGP synchronization coordinates LDP and IGP so that IGP advertises links with regular metrics only when MPLS LDP is converged on that link. LDP considers a link converged when at least one LDP session is up and running on the link for which LDP has sent its applicable label bindings and received at least one label binding from the peer. LDP communicates this information to IGP upon link up or session down events and IGP acts accordingly, depending on sync state.

LDP-IGP synchronization is supported for both OSPF and ISIS protocols and is configured under the corresponding IGP protocol configuration mode. Under certain circumstances, it might be required to delay declaration of re-synchronization to a configurable interval. LDP provides a configuration option to delay declaring synchronization up for up to 60 seconds. LDP communicates this information to IGP upon linkup or session down events.

Configuring LDP IGP Synchronization: Open Shortest Path First (OSPF) Example

This example shows how to configure LDP-IGP synchronization for an OSPF instance. The synchronization delay is configured as 30 seconds.

```
RP/0/RP0/CPU0:Router(config)# router ospf 100
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp sync
RP/0/RP0/CPU0:Router(config-ospf)# mpls ldp igp sync delay 30
RP/0/RP0/CPU0:Router(config-ospf)# commit
```
Configuring LDP IGP Synchronization: Intermediate System to Intermediate System (IS-IS)

This example shows how to configure LDP-IGP synchronization for IS-IS.

```
RP/0/RP0/CPU0:Router(config)# router isis 100
RP/0/RP0/CPU0:Router(config-isis)# interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:Router(config-isis-if-af)# mpls ldp sync
RP/0/RP0/CPU0:Router(config-isis-if-af)# commit
```

Configuring Label Distribution Protocol Graceful Restart

LDP Graceful Restart provides a mechanism for LDP peers to preserve the MPLS forwarding state when the LDP session goes down. Without LDP Graceful Restart, when an established session fails, the corresponding forwarding states are cleaned immediately from the restart and peer nodes. In this case, LDP forwarding has to restart from the beginning, causing a potential loss of data and connectivity. If LDP graceful restart is configured, traffic can continue to be forwarded without interruption, even when the LDP session restarts. The LDP graceful restart capability is negotiated between two peers during session initialization time. During session initialization, a router advertises its ability to perform LDP graceful restart by sending the graceful restart typed length value (TLV). This TLV contains the reconnect time and recovery time. The values of the reconnect and recovery times indicate the graceful restart capabilities supported by the router. The reconnect time is the amount of time the peer router waits for the restarting router to establish a connection. When a router discovers that a neighboring router is restarting, it waits until the end of the recovery time before attempting to reconnect. Recovery time is the amount of time that a neighboring router maintains its information about the restarting router.

Configuration Example

This example shows how to configure LDP graceful restart. In this example, the amount of time that a neighboring router maintains the forwarding state about the gracefully restarting router is specified as 180 seconds. The reconnect time is configured as 169 seconds.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# interface HundredGigE 0/0/0/5
RP/0/RP0/CPU0:Router(config-ldp-if)# exit
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart forwarding-state-holdtime 180
RP/0/RP0/CPU0:Router(config-ldp)# graceful-restart reconnect-timeout 169
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

Configuring Label Distribution Protocol Nonstop Routing

LDP nonstop routing (NSR) functionality makes failures, such as Route Processor (RP) or Distributed Route Processor (DRP) failover, invisible to routing peers with minimal to no disruption of convergence performance. By default, NSR is globally enabled on all LDP sessions except AToM.

A disruption in service may include any of these events:

- Route processor (RP) or distributed route processor (DRP) failover
- LDP process restart
- Minimum disruption restart (MDR)
Unlike graceful restart functionality, LDP NSR does not require protocol extensions and does not force software upgrades on other routers in the network, nor does LDP NSR require peer routers to support NSR. L2VPN configuration is not supported on NSR. Process failures of active LDP results in session loss and, as a result, NSR cannot be provided unless RP switchover is configured as a recovery action.

**Configuration Example**

This example shows how to configure LDP Non-Stop Routing.

```
RP/0/RP0/CPU0:Router(config)# mpls ldp
RP/0/RP0/CPU0:Router(config-ldp)# nsr
RP/0/RP0/CPU0:Router(config-ldp)# commit
```

**Verification**

```
RP/0/RP0/CPU0:Router# show mpls ldp nsr summary
Mon Dec 7 04:02:16.259 UTC
Sessions:
Total: 1, NSR-eligible: 1, Sync-ed: 0
(1 Ready)
```

**MPLS Label Distribution Protocol : Details**

This section provides detailed conceptual information about setting up LSPs, LDP graceful restart, and LDP session protection.

**Setting Up Label Switched Paths**

MPLS packets are forwarded between the nodes on the MPLS network using Label Switched Paths (LSPs). LSPs can be created statically or by using a label distribution protocol like LDP. Label Switched Paths created by LDP performs hop-by-hop path setup instead of an end-to-end path. LDP enables label switched routers (LSRs) to discover their potential peer routers and to establish LDP sessions with those peers to exchange label binding information.
The following figure illustrates the process of label binding exchange for setting up LSPs.

**Figure 1: Setting Up Label Switched Paths**

For a given network (10.0.0.0), hop-by-hop LSPs are set up between each of the adjacent routers (or, nodes) and each node allocates a local label and passes it to its neighbor as a binding:

1. R4 allocates local label L4 for prefix 10.0.0.0 and advertises it to its neighbors (R3).
2. R3 allocates local label L3 for prefix 10.0.0.0 and advertises it to its neighbors (R1, R2, R4).
3. R1 allocates local label L1 for prefix 10.0.0.0 and advertises it to its neighbors (R2, R3).
4. R2 allocates local label L2 for prefix 10.0.0.0 and advertises it to its neighbors (R1, R3).
5. R1’s label information base (LIB) keeps local and remote labels bindings from its neighbors.
6. R2's LIB keeps local and remote labels bindings from its neighbors.
7. R3’s LIB keeps local and remote labels bindings from its neighbors.
8. R4’s LIB keeps local and remote labels bindings from its neighbors.
MPLS Forwarding

Once the label bindings are learned, MPLS forwarding plane is setup and packets are forwarded as shown in the following figure.

Figure 2: MPLS Forwarding

1. Because R3 is next hop for 10.0.0.0 as notified by the FIB, R1 selects label binding from R3 and installs forwarding entry (Layer 1, Layer 3).

2. Because R3 is next hop for 10.0.0.0 (as notified by FIB), R2 selects label binding from R3 and installs forwarding entry (Layer 2, Layer 3).

3. Because R4 is next hop for 10.0.0.0 (as notified by FIB), R3 selects label binding from R4 and installs forwarding entry (Layer 3, Layer 4).

4. Because next hop for 10.0.0.0 (as notified by FIB) is beyond R4, R4 uses NO-LABEL as the outbound and installs the forwarding entry (Layer 4); the outbound packet is forwarded IP-only.

5. Incoming IP traffic on ingress LSR R1 gets label-imposed and is forwarded as an MPLS packet with label L3.

6. Incoming IP traffic on ingress LSR R2 gets label-imposed and is forwarded as an MPLS packet with label L3.

7. R3 receives an MPLS packet with label L3, looks up in the MPLS label forwarding table and switches this packet as an MPLS packet with label L4.

8. R4 receives an MPLS packet with label L4, looks up in the MPLS label forwarding table and finds that it should be Unlabeled, pops the top label, and passes it to the IP forwarding plane.

9. IP forwarding takes over and forwards the packet onward.
Details of Label Distribution Protocol Graceful Restart

LDP (Label Distribution Protocol) graceful restart provides a control plane mechanism to ensure high availability and allows detection and recovery from failure conditions while preserving Nonstop Forwarding (NSF) services. Graceful restart is a way to recover from signaling and control plane failures without impacting forwarding.

Without LDP graceful restart, when an established session fails, the corresponding forwarding states are cleaned immediately from the restarting and peer nodes. In this case LDP forwarding restarts from the beginning, causing a potential loss of data and connectivity.

The LDP graceful restart capability is negotiated between two peers during session initialization time, in FT SESSION TLV. In this typed length value (TLV), each peer advertises the following information to its peers:

**Reconnect time**
- Advertises the maximum time that other peer will wait for this LSR to reconnect after control channel failure.

**Recovery time**
- Advertises the maximum time that the other peer has on its side to reinstate or refresh its states with this LSR. This time is used only during session reestablishment after earlier session failure.

**FT flag**
- Specifies whether a restart could restore the preserved (local) node state for this flag.

Once the graceful restart session parameters are conveyed and the session is up and running, graceful restart procedures are activated.

When configuring the LDP graceful restart process in a network with multiple links, targeted LDP hello adjacencies with the same neighbor, or both, make sure that graceful restart is activated on the session before any hello adjacency times out in case of neighbor control plane failures. One way of achieving this is by configuring a lower session hold time between neighbors such that session timeout occurs before hello adjacency timeout. It is recommended to set LDP session hold time using the following formula:

\[
\text{Session Holdtime} \leq (\text{Hello holdtime} - \text{Hello interval}) \times 3
\]

This means that for default values of 15 seconds and 5 seconds for link Hello holdtime and interval respectively, session hold time should be set to 30 seconds at most.

**Phases in Graceful Restart**

The graceful restart mechanism is divided into different phases:

**Control communication failure detection**
- Control communication failure is detected when the system detects either:
  - Missed LDP hello discovery messages
  - Missed LDP keepalive protocol messages
  - Detection of Transmission Control Protocol (TCP) disconnection a with a peer
Forwarding state maintenance during failure

Persistent forwarding states at each LSR are achieved through persistent storage (checkpoint) by the LDP control plane. While the control plane is in the process of recovering, the forwarding plane keeps the forwarding states, but marks them as stale. Similarly, the peer control plane also keeps (and marks as stale) the installed forwarding rewrites associated with the node that is restarting. The combination of local node forwarding and remote node forwarding plane states ensures NSF and no disruption in the traffic.

Control state recovery

Recovery occurs when the session is reestablished and label bindings are exchanged again. This process allows the peer nodes to synchronize and to refresh stale forwarding states.

Control Plane Failure

When a control plane failure occurs, connectivity can be affected. The forwarding states installed by the router control planes are lost, and the in-transit packets could be dropped, thus breaking NSF. The following figure illustrates control plane failure and recovery with graceful restart and shows the process and results of a control plane failure leading to loss of connectivity and recovery using graceful restart.

Figure 3: Control Plane Failure
Recovery with Graceful Restart

**Figure 4: Recovering with Graceful Restart**

1. The R4 LSR control plane restarts.
2. L4B is lost when the control plane restarts.
3. The forwarding states installed by the R4 LDP control plane are immediately deleted.
4. Any in-transit packets flowing from R3 to R4 (still labeled with L4) arrive at R4.
5. The MPLS forwarding plane at R4 performs a lookup on local label L4 which fails. Because of this failure, the packet is dropped and NSF is not met.
6. The R3 LDP peer detects the failure of the control plane channel and deletes its label bindings from R4.
7. The R3 control plane stops using outgoing labels from R4 and deletes the corresponding forwarding state (rewrites), which in turn causes forwarding disruption.
8. The established LSPs connected to R4 are terminated at R3, resulting in broken end-to-end LSPs from R1 to R4.
9. The established LSPs connected to R4 are terminated at R3, resulting in broken LSPs end-to-end from R2 to R4.

When the LDP control plane recovers, the restarting LSR starts its forwarding state hold timer and restores its forwarding state from the checkpointed data. This action reinstates the forwarding state and entries and marks them as old.

The restarting LSR reconnects to its peer, indicated in the FT Session TLV, that it either was or was not able to restore its state successfully. If it was able to restore the state, the bindings are resynchronized.

The peer LSR stops the neighbor reconnect timer (started by the restarting LSR), when the restarting peer connects and starts the neighbor recovery timer. The peer LSR checks the FT Session TLV if the restarting
peer was able to restore its state successfully. It reinstates the corresponding forwarding state entries and receives binding from the restarting peer. When the recovery timer expires, any forwarding state that is still marked as stale is deleted.

If the restarting LSR fails to recover (restart), the restarting LSR forwarding state and entries will eventually timeout and is deleted, while neighbor-related forwarding states or entries are removed by the Peer LSR on expiration of the reconnect or recovery timers.

**Details of Session Protection**

LDP session protection lets you configure LDP to automatically protect sessions with all or a given set of peers (as specified by peer-acl). When configured, LDP initiates backup targeted hellos automatically for neighbors for which primary link adjacencies already exist. These backup targeted hellos maintain LDP sessions when primary link adjacencies go down.

The Session Protection figure illustrates LDP session protection between neighbors R1 and R3. The primary link adjacency between R1 and R3 is directly connected link and the backup; targeted adjacency is maintained between R1 and R3. If the direct link fails, LDP link adjacency is destroyed, but the session is kept up and running using targeted hello adjacency (through R2). When the direct link comes back up, there is no change in the LDP session state and LDP can converge quickly and begin forwarding MPLS traffic.

*Figure 5: Session Protection*

When LDP session protection is activated (upon link failure), protection is maintained for an unlimited period time.
Implementing MPLS Static Labeling

MPLS Static Labeling

The MPLS static feature enables you to statically assign local labels to an IPv4 prefix. Also, Label Switched Paths (LSPs) can be provisioned for these static labels by specifying the next-hop information that is required to forward the packets containing static label.

If there is any discrepancy between labels assigned statically and dynamically, the router issues a warning message in the console log. By means of this warning message, the discrepancy can be identified and resolved.

The advantages of static labels over dynamic labels are:

- Improve security because the risk of receiving unwanted labels from peers (running a compromised MPLS dynamic labeling protocol) is reduced.
- Gives users full control over defined LSPs.
- Utilize system resources optimally because dynamic labeling is not processed.

Restrictions

- Static labeling on IPv6 packets is not supported.
- The router does not prevent label discrepancy at the time of configuring static labels. Any generated discrepancy needs to be subsequently cleared.
- Equal-cost multi-path routing (ECMP) is not supported.
- Interfaces must be explicitly configured to handle traffic with static MPLS labels.
- The MPLS per-VRF labels cannot be shared between MPLS static and other applications.

Define Label Range and Enable MPLS Encapsulation

By default, MPLS encapsulation is disabled on all interfaces. MPLS encapsulation has to be explicitly enabled on all ingress and egress MPLS interfaces through which the static MPLS labeled traffic travels.
Also, the dynamic label range needs to be defined. Any label that falls outside this dynamic range is available for manually allocating as static labels. The router does not verify statically-configured labels against the specified label range. Therefore, to prevent label discrepancy, ensure that you do not configure static MPLS labels that fall within the dynamic label range.

**Configuration Example**

You have to accomplish the following to complete the MPLS static labeling configuration. Values are provided as an example.

1. Define a dynamic label range, which in this task is set between 17000 and 18000.
2. Enable MPLS encapsulation on the required interface.
3. Setup a static MPLS LSP for a specific ingress label 24035.
4. Specify the forwarding information so that for packets that are received with the label, 24035, the MPLS protocol swaps labels and applies the label, 24036. After applying the new label, it forwards the packets to the next hop, 10.2.2.2, through the specified interface.

```
RP/0/RP0/CPU0:router(config)# mpls label range table 0 17000 18000
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# mpls static
RP/0/RP0/CPU0:router(config-mpls-static)# interface HundredGigE 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-static)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-mpls-static-af)# local-label 24035 allocate
RP/0/RP0/CPU0:router(config-mpls-static-af-lbl)# forward
RP/0/RP0/CPU0:router(config-mpls-static-af-lbl-fwd)# path 1 nexthop HundredGigE 0/0/0/1 10.2.2.2 out-label 24036
RP/0/RP0/CPU0:router(config-mpls-static-af-lbl-fwd)# commit
```

**Verification**

Verify the interfaces on which MPLS is enabled.

```
RP/0/RP0/CPU0:router# show mpls interfaces
```

Verify that the status is "Created" for the specified label value.

```
RP/0/RP0/CPU0:router#show mpls static local-label all
```

Check the dynamic range and ensure that the specified local-label value is outside this range.

```
RP/0/RP0/CPU0:router#show mpls label range
```

Verify that the MPLS static configuration has taken effect, and the label forwarding is taking place.

```
RP/0/RP0/CPU0:router#show mpls lsd forwarding
```
Associated Commands

- mpls static
- mpls label range
- show mpls interfaces

Identify and Clear Label Discrepancy

During configuring or de-configuring static labels or a label range, a label discrepancy can get generated when:

- A static label is configured for an IP prefix that already has a binding with a dynamic label.
- A static label is configured for an IP prefix, when the same label value is dynamically allocated to another IP prefix.

Verification

Identify label discrepancy by using these show commands.

Router# show mpls static local-label discrepancy
Tue Apr 22 18:36:31.614 UTC
Label VRF Type Prefix RW Configured Status
------- --------------- ------------ ---------------- --------------- --------
24000 default X-Connect NA Yes

Discrepancy

Router# show mpls static local-label all
Tue Apr 22 18:36:31.614 UTC
Label VRF Type Prefix RW Configured Status
------- --------------- ------------ ---------------- --------------- --------
24000 default X-Connect N/A Yes
24035 default X-Connect N/A Yes

Discrepancy

RP/0/RP0/CPU0# show log
Thu Apr 24 14:18:53.743 UTC
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
Console logging: level warnings, 199 messages logged
Monitor logging: level debugging, 0 messages logged
Trap logging: level informational, 0 messages logged
Buffer logging: level debugging, 2 messages logged

Log Buffer (307200 bytes):

RP/0/RP0/CPU0# Apr 24 14:18:53.743 : mpls_static[1043]:

ROUTING-MPLS_STATIC-7-ERR_STATIC_LABEL_DISCREPANCY
The system detected 1 label discrepancies (static label could not be allocated due to conflict with other applications).
Please use 'clear mpls static local-label discrepancy' to fix this issue.

RP/0/RP0/CPU0# Apr 24 14:18:53.937 : config[65762]: %MGBL-CONFIG-6-DB_COMMIT: Configuration committed by user 'cisco'.
Use 'show configuration commit changes 1000000020' to view the changes.
Rectification

Label discrepancy is cleared by allocating a new label to those IP prefixes that are allocated dynamic label. The static label configuration takes precedence while clearing discrepancy. Traffic can be affected while clearing discrepancy.

```
Router# clear mpls static local-label discrepancy all
```

Verify that the discrepancy is cleared.

```
Router# show mpls static local-label all

Wed Nov 25 21:45:50.368 UTC
Label VRF Type Prefix RW Configured Status
------- --------------- ------------ ---------------- --------------- --------
24000 default X-Connect N/A Yes Created
24035 default X-Connect N/A Yes Created
```

Associated Commands

- show mpls static local-label discrepancy
- clear mpls static local-label discrepancy all
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 an LSP from router R1 through router R4 that can be used for TE in an MPLS environment.

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
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, on page 28.

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:

- ncs5500-mpls-2.0.0.0-r601.x86_64.rpm-6.0.1
- ncs5500-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)# 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: 502396459702858020 (0x45b8b99b00000124)
Messages successfully authenticated: 245
Messages failed authentication: 0
```

Related Topics

- Configuring RSVP Authentication for an Interface, on page 23
- Configuring RSVP Authentication on a Neighbor, on page 24
- RSVP for MPLS-TE Features- Details, on page 28

Configuring RSVP Authentication for an Interface

You can individually configure the values of RSVP authentication parameters including key-chain, lifetime, 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, lifetime 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 22
• Configuring RSVP Authentication on a Neighbor, on page 24
• RSVP for MPLS-TE Features- Details, on page 28

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
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
```
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 IP header. 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.

```plaintext
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 ip host 1.1.1.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ip 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 rsvpac1
RP/0/RP0/CPU0:router(config)# commit
```

**Verification**

Verify the configuration of ACL based prefix filtering

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

**Related Topics**

- Configuring RSVP Packet Dropping, on page 27

**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.

```plaintext
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 ip host 1.1.1.1 any
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny ip 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
RP/0/RP0/CPU0:router(config)# commit
```

**Verification**

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

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

- Configuring ACL Based Prefix Filtering, on page 26

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 send 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 Table 1: Security Association Parameters, on page 29

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

<table>
<thead>
<tr>
<th>Security Association Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>src</td>
<td>IP address of the sender.</td>
</tr>
<tr>
<td>dst</td>
<td>IP address of the final destination.</td>
</tr>
<tr>
<td>interface</td>
<td>Interface of the security association.</td>
</tr>
<tr>
<td>direction</td>
<td>Send or receive type of the security association.</td>
</tr>
<tr>
<td>Lifetime</td>
<td>Expiration timer value that is used to collect unused security association data.</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>Last sequence number that was either sent or accepted (dependent of the direction type).</td>
</tr>
<tr>
<td>key-source</td>
<td>Source of keys for the configurable parameter.</td>
</tr>
<tr>
<td>keyID</td>
<td>Key number (returned form the key-source) that was last used.</td>
</tr>
<tr>
<td>Security Association Parameters</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Window Size</td>
<td>Specifies the maximum number of authenticated messages that can be received out of order.</td>
</tr>
<tr>
<td>Window</td>
<td>Specifies the last window size value sequence number that is received or accepted.</td>
</tr>
</tbody>
</table>
Implementing MPLS Traffic Engineering

Implementing MPLS Traffic Engineering

Traditional IP routing emphasizes on forwarding traffic to the destination as fast as possible. As a result, the routing protocols find out the least-cost route according to its metric to each destination in the network and every router forwards the packet based on the destination IP address and packets are forwarded hop-by-hop. Thus, traditional IP routing does not take into account the available bandwidth of the link. This can cause some links to be over-utilized compared to others and bandwidth is not efficiently utilized. Traffic Engineering (TE) is used when the problems result from inefficient mapping of traffic streams onto the network resources. Traffic engineering allows you to control the path that data packets follow and moves traffic flows from congested links to non-congested links that would not be possible by the automatically computed destination-based shortest path.

Multiprotocol Label Switching (MPLS) with its label switching capabilities, eliminates the need for an IP route look-up and creates a virtual circuit (VC) switching function, allowing enterprises the same performance on their IP-based network services as with those delivered over traditional networks such as Frame Relay or Asynchronous Transfer Mode (ATM). MPLS traffic engineering (MPLS-TE) relies on the MPLS backbone to replicate and expand upon the TE capabilities of Layer 2 ATM and Frame Relay networks.

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. The point where the tunnel begins is called the tunnel head-end or tunnel source, and the node where the tunnel ends is called the tunnel tail-end or tunnel destination.

MPLS uses extensions to a link-state based Interior Gateway Protocol (IGP), such as Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF). MPLS calculates TE tunnels at the LSP head based on required and available resources (constraint-based routing). If configured, the IGP automatically routes the traffic onto these LSPs. Typically, a packet that crosses the MPLS-TE backbone travels on a single LSP that connects the ingress point to the egress point. MPLS TE automatically establishes and maintains the LSPs across the MPLS network by using the Resource Reservation Protocol (RSVP).
Overview of MPLS-TE Features

In MPLS traffic engineering, IGP extensions flood the TE information across the network. Once the IGP distributes the link attributes and bandwidth information, the head-end router calculates the best path from head to tail for the MPLS-TE tunnel. This path can also be configured explicitly. Once the path is calculated, RSVP-TE is used to setup the TE LSP (Labeled Switch Path).

To forward the traffic, you can configure autoroute or forward adjacency. The autoroute feature will announce the routes assigned by the tail-end router and its downstream routes to the routing table of the head end router and the tunnel is considered as a directly connected link to the tunnel.

If forward adjacency is enabled, MPLS-TE tunnel is advertised as a link in an IGP network with the link's cost associated with it. Routers outside of the TE domain can see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network.

MPLS-TE provides protection mechanism known as fast re-route to minimize packet loss during a failure. For fast reroute, you need to create backup tunnels. The autotunnel backup feature enables a router to dynamically build backup tunnels when they are needed instead of pre-configuring each backup tunnel and then assign the backup tunnel to the protected interfaces.

DiffServ Aware Traffic Engineering (DS-TE) provides the ability to configure multiple bandwidth constraints on an MPLS-enabled interface to support various classes of service (CoS). These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint.

The MPLS traffic engineering auto-tunnel mesh feature allows you to set up full mesh of TE tunnels automatically with a minimal set of MPLS traffic engineering configurations. The MPLS-TE auto bandwidth feature allows to automatically adjusts bandwidth based on traffic patterns without traffic disruption.

The MPLS-TE interarea tunneling feature allows you to establish TE tunnels spanning multiple Interior Gateway Protocol (IGP) areas and levels, thereby eliminating the requirement that head-end and tail-end routers should reside in a single area.

For detailed information about MPLS-TE features, see MPLS-TE Features - Details, on page 44

How MPLS-TE Works

MPLS-TE automatically establishes and maintains label switched paths (LSPs) across the backbone by using RSVP. The path that an LSP uses is determined by the LSP resource requirements and network resources, such as bandwidth. Available resources are flooded by means of extensions to a link-state-based Interior Gateway Protocol (IGP). MPLS-TE tunnels are calculated at the LSP head-end router, based on a fit between the required and available resources (constraint-based routing). The IGP automatically routes the traffic to these LSPs. Typically, a packet crossing the MPLS-TE backbone travels on a single LSP that connects the ingress point to the egress point.

The following sections describe the components of MPLS-TE:

Tunnel Interfaces

From a Layer 2 standpoint, an MPLS tunnel interface represents the head-end of an LSP. It is configured with a set of resource requirements, such as bandwidth and media requirements, and priority. From a Layer 3 standpoint, an LSP tunnel interface is the head-end of a unidirectional virtual link to the tunnel destination.
**MPLS-TE Path Calculation Module**

This calculation module operates at the LSP head-end. The module determines a path to use for an LSP. The path calculation uses a link-state database containing flooded topology and resource information.

**RSVP with TE Extensions**

RSVP operates at each LSP hop and is used to signal and maintain LSPs based on the calculated path.

**MPLS-TE Link Management Module**

This module operates at each LSP hop, performs link call admission on the RSVP signaling messages, and keep track on topology and resource information to be flooded.

**Link-state IGP**

Either Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF) can be used as IGPs. These IGPs are used to globally flood topology and resource information from the link management module.

**Label Switching Forwarding**

This forwarding mechanism provides routers with a Layer 2-like ability to direct traffic across multiple hops of the LSP established by RSVP signaling.

---

**Configuring MPLS-TE**

MPLS-TE requires co-ordination among several global neighbor routers. RSVP, MPLS-TE and IGP are configured on all routers and interfaces in the MPLS traffic engineering network. Explicit path and TE tunnel interfaces are configured only on the head-end routers. MPLS-TE requires some basic configuration tasks explained in this section.

**Building MPLS-TE Topology**

Building MPLS-TE topology, sets up the environment for creating MPLS-TE tunnels. This procedure includes the basic node and interface configuration for enabling MPLS-TE. To perform constraint based routing, OSPF or IS-IS needs to be enabled as IGP extension.

**Before You Begin**

Before you start to build the MPLS-TE topology, the following pre-requisites are required:

- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

- Enable RSVP on the port interface.
Example
This example enables MPLS-TE on a node and then specify the interface that is part of the MPLS-TE. Here, OSPF is used as the IGP extension protocol for information distribution.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config)# router ospf area 1
RP/0/RP0/CPU0:router(config-router)# area 0
RP/0/RP0/CPU0:router(config-ospf-ar)# exit
RP/0/RP0/CPU0:router(config-ospf)# mpls traffic-eng router-id 192.168.70.1
RP/0/RP0/CPU0:router(config)# commit
```

Example
This example enables MPLS-TE on a node and then specify the interface that is part of the MPLS-TE. Here, IS-IS is used as the IGP extension protocol for information distribution.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# net 47.0001.0000.0000.0002.00
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide
RP/0/RP0/CPU0:router(config-isis-af)# mpls traffic-eng level 1
RP/0/RP0/CPU0:router(config)# commit
```

Related Topics
- How MPLS-TE Works, on page 32
- Creating an MPLS-TE Tunnel, on page 34

Creating an MPLS-TE Tunnel
Creating an MPLS-TE tunnel is a process of customizing the traffic engineering to fit your network topology. The MPLS-TE tunnel is created at the head-end router. You need to specify the destination and path of the TE LSP.

Before you Begin
The following prerequisites are required to create an MPLS-TE tunnel:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example
This example configures a MPLS-TE tunnel on the head-end router with a destination IP address 192.168.92.125. The bandwidth for the tunnel, path-option and forwarding parameters of the tunnel are also configured.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
```
RP/0/RP0/CPU0:router(config-if)# path-option 1 dynamic
RP/0/RP0/CPU0:router(config-if)# autoroute announce or forwarding adjacency
RP/0/RP0/CPU0:router(config-if)# signalled-bandwidth 100
RP/0/RP0/CPU0:router(config)# commit

Verification

Verify the configuration of MPLS-TE tunnel using the following command.

RP/0/RP0/CPU0:router# show mpls traffic-engineering tunnels brief

<table>
<thead>
<tr>
<th>Signalling Summary:</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSP Tunnels Process: running</td>
</tr>
<tr>
<td>RSVP Process: running</td>
</tr>
<tr>
<td>Forwarding: enabled</td>
</tr>
<tr>
<td>Periodic reoptimization: every 3600 seconds, next in 2538 seconds</td>
</tr>
<tr>
<td>Periodic FRR Promotion: every 300 seconds, next in 38 seconds</td>
</tr>
<tr>
<td>Auto-bw enabled tunnels: 0 (disabled)</td>
</tr>
<tr>
<td>TUNNEL NAME</td>
</tr>
<tr>
<td>tunnel-te1</td>
</tr>
</tbody>
</table>

Displayed 1 up, 0 down, 0 recovering, 0 recovered heads

Related Topics

- How MPLS-TE Works, on page 32
- Building MPLS-TE Topology, on page 33

Configuring Fast Reroute

Fast reroute (FRR) provides link protection to LSPs enabling the traffic carried by LSPs that encounter a failed link to be rerouted around the failure. The reroute decision is controlled locally by the router connected to the failed link. The head-end router on the tunnel is notified of the link failure through IGP or through RSVP. When it is notified of a link failure, the head-end router attempts to establish a new LSP that bypasses the failure. This provides a path to reestablish links that fail, providing protection to data transfer.

Before you Begin

The following prerequisites are required to create an MPLS-TE tunnel:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures fast reroute on a MPLS-TE tunnel. Here, tunnel-te 2 is configured as the back-up tunnel.

RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# fast-reroute
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# backup-path tunnel-te 2
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
**Configuring MPLS-TE**

```
RP/0/RP0/CPU0:router(config-if)# backup-bw global-pool 5000
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# path-option l explicit name backup-path
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config)# commit
```

**Verification**

Use the `show mpls traffic-eng fast-reroute database` command to verify the fast reroute configuration.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng fast-reroute database
```

<table>
<thead>
<tr>
<th>Tunnel head FRR information:</th>
<th>FRR intf/label</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>tt4000 HundredGigabitEthernet 0/0/0/3:34 tt1000:34</td>
<td>Ready</td>
<td></td>
</tr>
<tr>
<td>tt4001 HundredGigabitEthernet 0/0/0/3:35 tt1001:35</td>
<td>Ready</td>
<td></td>
</tr>
<tr>
<td>tt4002 HundredGigabitEthernet 0/0/0/3:36 tt1001:36</td>
<td>Ready</td>
<td></td>
</tr>
</tbody>
</table>

**Related Topics**

- [Configuring MPLS-TE](#), on page 33
- [Configuring AutoTunnel Backup](#), on page 36
- [Configuring Next Hop Backup Tunnel](#), on page 37
- [MPLS-TE Features - Details](#), on page 44

### Configuring AutoTunnel Backup

The MPLS Traffic Engineering AutoTunnel Backup feature enables a router to dynamically build backup tunnels on the interfaces that are configured with MPLS TE tunnels instead of building MPLS-TE tunnels statically.

The MPLS-TE AutoTunnel Backup feature has these benefits:

- Backup tunnels are built automatically, eliminating the need for users to pre-configure each backup tunnel and then assign the backup tunnel to the protected interface.
- Protection is expanded—FRR does not protect IP traffic that is not using the TE tunnel or Label Distribution Protocol (LDP) labels that are not using the TE tunnel.

The TE attribute-set template that specifies a set of TE tunnel attributes, is locally configured at the head-end of auto-tunnels. The control plane triggers the automatic provisioning of a corresponding TE tunnel, whose characteristics are specified in the respective attribute-set.

### Configuration Example

This example configures AutoTunnel backup on an interface and specifies the attribute-set template for the auto tunnels. In this example, unused backup tunnels are removed every 20 minutes using a timer and also the range of tunnel interface numbers are specified.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# auto-tunnel backup
RP/0/RP0/CPU0:router(config-mpls-te-if-auto-backup)# attribute-set ab
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup timers removal unused 20
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup tunnel-id min 6000 max 6500
RP/0/RP0/CPU0:router(config)# commit
```
Verification

This example shows a sample output for automatic backup tunnel configuration.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels brief

<table>
<thead>
<tr>
<th>TUNNEL NAME</th>
<th>DESTINATION</th>
<th>STATUS</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>tunnel-te0</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>tunnel-te1</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>tunnel-te2</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>tunnel-te50</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>*tunnel-te60</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>*tunnel-te70</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>*tunnel-te80</td>
<td>200.0.0.3</td>
<td>up</td>
<td>up</td>
</tr>
</tbody>
</table>
```

Related Topics

- Configuring Fast Reroute, on page 35
- Configuring Next Hop Backup Tunnel, on page 37
- MPLS-TE Features - Details, on page 44

Configuring Next Hop Backup Tunnel

The backup tunnels that bypass only a single link of the LSP path are referred as Next Hop (NHOP) backup tunnels because they terminate at the LSP's next hop beyond the point of failure. They protect LSPs, if a link along their path fails, by rerouting the LSP traffic to the next hop, thereby bypassing the failed link.

Configuration Example

This example configures next hop backup tunnel on an interface and specifies the attribute-set template for the auto tunnels. In this example, unused backup tunnels are removed every 20 minutes using a timer and also the range of tunnel interface numbers are specified.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# auto-tunnel backup nhop-only
RP/0/RP0/CPU0:router(config-mpls-te-if-auto-backup)# attribute-set ab
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup timers removal unused 20
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel backup tunnel-id min 6000 max 6500
RP/0/RP0/CPU0:router(config)# commit
```

Related Topics

- Configuring AutoTunnel Backup, on page 36
- Configuring Fast Reroute, on page 35
- MPLS-TE Features - Details, on page 44

Configuring SRLG Node Protection

Shared Risk Link Groups (SRLG) in MPLS traffic engineering refer to situations in which links in a network share common resources. These links have a shared risk, and that is when one link fails, other links in the group might fail too.
OSPF and Intermediate System-to-Intermediate System (IS-IS) flood the SRLG value information (including other TE link attributes such as bandwidth availability and affinity) using a sub-type length value (sub-TLV), so that all routers in the network have the SRLG information for each link.

MPLS-TE SRLG feature enhances backup tunnel path selection by avoiding using links that are in the same SRLG as the interfaces it is protecting while creating back up tunnels.

**Configuration Example**

This example creates a back up tunnel and excludes the protected node IP address from the explicit path.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# backup-path tunnl-te 2
RP/0/RP0/CPU0:router(config-mpls-te-if)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# path-option 1 explicit name backup-srlg
RP/0/RP0/CPU0:router(config-if)# destination 192.168.92.125
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# explicit-path name backup-srlg-nodep
RP/0/RP0/CPU0:router(config-if)# index 1 exclude-address 192.168.91.1
RP/0/RP0/CPU0:router(config-if)# index 1 exclude-srlg 192.168.92.2
RP/0/RP0/CPU0:router(config)# commit
```

**Related Topics**

- Configuring Fast Reroute, on page 35
- MPLS-TE Features - Details, on page 44

**Configuring Pre-Standard DS-TE**

Regular traffic engineering does not provide bandwidth guarantees to different traffic classes. A single bandwidth constraint is used in regular TE that is shared by all traffic. MPLS DS-TE provides the ability to configure multiple bandwidth constraints on an MPLS-enabled interface. These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint. Cisco IOS XR software supports two DS-TE modes: Pre-standard and IETF. Pre-standard DS-TE uses the Cisco proprietary mechanisms for RSVP signaling and IGP advertisements. This DS-TE mode does not interoperate with third-party vendor equipment. Pre-standard DS-TE is enabled only after configuring the sub-pool bandwidth values on MPLS-enabled interfaces.

Pre-standard Diff-Serve TE mode supports a single bandwidth constraint model a Russian Doll Model (RDM) with two bandwidth pools: global-pool and sub-pool.

**Before you Begin**

The following prerequisites are required to configure a Pre-standard DS-TE tunnel.

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.
Configuration Example

This example configures a pre-standard DS-TE tunnel.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth 100 150 sub-pool 50
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10
RP/0/RP0/CPU0:router(config)# commit
```

Verification

Use the `show mpls traffic-eng topology` command to verify the pre-standard DS-TE tunnel configuration.

Related Topics

- Configuring an IETF DS-TE Tunnel Using RDM, on page 39
- Configuring an IETF DS-TE Tunnel Using MAM, on page 40
- MPLS-TE Features - Details, on page 44

Configuring an IETF DS-TE Tunnel Using RDM

IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode interoperates with third-party vendor equipment.

IETF mode supports multiple bandwidth constraint models, including RDM and MAM, both with two bandwidth pools. In an IETF DS-TE network, identical bandwidth constraint models must be configured on all nodes.

Before you Begin

The following prerequisites are required to create an IETF mode DS-TE tunnel using RDM:

- You must have a router ID for the neighboring router.
- Stable router ID is required at either end of the link to ensure that the link is successful. If you do not assign a router ID to the routers, the system defaults to the global router ID. Default router IDs are subject to change, which can result in an unstable link.

Configuration Example

This example configures an IETF DS-TE tunnel using RDM.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth rdm 100 150
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf
RP/0/RP0/CPU0:router(config-mpls-te)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10 class-type 1
RP/0/RP0/CPU0:router(config)# commit
```
Verification
Use the show mpls traffic-eng topology command to verify the IETF DS-TE tunnel using RDM configuration.

Related Topics
- Configuring Pre-Standard DS-TE, on page 38
- Configuring an IETF DS-TE Tunnel Using MAM, on page 40
- MPLS-TE Features - Details, on page 44

Configuring an IETF DS-TE Tunnel Using MAM
IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode interoperate with third-party vendor equipment. IETF mode supports multiple bandwidth constraint models, including RDM and Maximum Allocation Model (MAM), both with two bandwidth pools.

Configuration Example
This example configures an IETF DS-TE tunnel using MAM.

```
RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# rsvp interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-rsvp-if)# bandwidth mam max-reservable-bw 1000 bc0 600 bc1 400
RP/0/RP0/CPU0:router(config-rsvp-if)# exit
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te mode ietf
RP/0/RP0/CPU0:router(config-mpls-te)# ds-te bc-model mam
RP/0/RP0/CPU0:router(config-mpls-te)# exit
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# signalled bandwidth sub-pool 10 class-type 1
RP/0/RP0/CPU0:router(config)# commit
```

Verification
Use the show mpls traffic-eng topology command to verify the IETF DS-TE tunnel using MAM configuration.

Related Topics
- Configuring an IETF DS-TE Tunnel Using RDM, on page 39
- Configuring Pre-Standard DS-TE, on page 38
- MPLS-TE Features - Details, on page 44

Configuring Flexible Name-Based Tunnel Constraints
MPLS-TE Flexible Name-based Tunnel Constraints provides a simplified and more flexible means of configuring link attributes and path affinities to compute paths for the MPLS-TE tunnels.

In the traditional TE, links are configured with attribute-flags that are flooded with TE link-state parameters using Interior Gateway Protocols (IGPs), such as Open Shortest Path First (OSPF).

MPLS-TE Flexible Name-based Tunnel Constraints lets you assign, or map, up to 32 color names for affinity and attribute-flag attributes instead of 32-bit hexadecimal numbers. After mappings are defined, the attributes can be referred to by the corresponding color name.
**Configuration Example**

This example shows assigning a how to associate a tunnel with affinity constraints.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# affinity-map red 1
RP/0/RP0/CPU0:router(config-mpls-te)# interface HundredGigabitEthernet 0/0/0/3
RP/0/RP0/CPU0:router(config-mpls-te-if)# attribute-names red
RP/0/RP0/CPU0:router(config)# interface tunnel-te 2
RP/0/RP0/CPU0:router(config-if)# affinity include red
RP/0/RP0/CPU0:router(config)# commit
```

**Configuring Automatic Bandwidth**

Automatic bandwidth allows to dynamically adjust bandwidth reservation based on measured traffic. MPLS-TE automatic bandwidth monitors the traffic rate on a tunnel interface and resizes the bandwidth on the tunnel interface to align it closely with the traffic in the tunnel. MPLS-TE automatic bandwidth is configured on individual Label Switched Paths (LSPs) at every head-end router.

The following table specifies the parameters that can be configured as part of automatic bandwidth configuration.

<table>
<thead>
<tr>
<th>Bandwidth Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application frequency</td>
<td>Configures how often the tunnel bandwidths changed for each tunnel. The default value is 24 hours.</td>
</tr>
<tr>
<td>Bandwidth limit</td>
<td>Configures the minimum and maximum automatic bandwidth to set on a tunnel.</td>
</tr>
<tr>
<td>Bandwidth collection frequency</td>
<td>Enables bandwidth collection without adjusting the automatic bandwidth. The default value is 5 minutes.</td>
</tr>
<tr>
<td>Overflow threshold</td>
<td>Configures tunnel overflow detection.</td>
</tr>
<tr>
<td>Adjustment threshold</td>
<td>Configures the tunnel-bandwidth change threshold to trigger an adjustment.</td>
</tr>
</tbody>
</table>

**Configuration Example**

This example enables automatic bandwidth on MPLS-TE tunnel interface and configure the following automatic bandwidth variables.

- Application frequency
- Bandwidth limit
- Adjustment threshold
• Overflow detection

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-te 1
RP/0/RP0/CPU0:router(config-if)# auto-bw
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# application 1000
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# bw-limit min 30 max 1000
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# adjustment-threshold 50 min 800
RP/0/RP0/CPU0:router(config-if-tunte-autobw)# overflow threshold 100 limit 1
RP/0/RP0/CPU0:router(config)# commit

Verification

Verify the automatic bandwidth configuration using the `show mpls traffic-eng tunnels auto-bw brief` command.

RP/0/RP0/CPU0:router# show mpls traffic-eng tunnels auto-bw brief

<table>
<thead>
<tr>
<th>Tunnel Name</th>
<th>LSP ID</th>
<th>Last appl BW (kbps)</th>
<th>Requested BW (kbps)</th>
<th>Signalled BW (kbps)</th>
<th>Highest BW (kbps)</th>
<th>Application Time Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>tunnel-te1</td>
<td>5</td>
<td>500</td>
<td>300</td>
<td>420</td>
<td></td>
<td>1h 10m</td>
</tr>
</tbody>
</table>

Related Topics
• MPLS-TE Features - Details, on page 44

Configuring AutoTunnel Mesh

The MPLS-TE auto-tunnel mesh (auto-mesh) feature allows you to set up full mesh of TE Point-to-Point (P2P) tunnels automatically with a minimal set of MPLS traffic engineering configurations. You can configure one or more mesh-groups and each mesh-group requires a destination-list (IPv4 prefix-list) listing destinations, which are used as destinations for creating tunnels for that mesh-group.

You can configure MPLS-TE auto-mesh type attribute-sets (templates) and associate them to mesh-groups. Label Switching Routers (LSRs) can create tunnels using the tunnel properties defined in this attribute-set.

Auto-Tunnel mesh configuration minimizes the initial configuration of the network. You can configure tunnel properties template and mesh-groups or destination-lists on each TE LSRs that further creates full mesh of TE tunnels between those LSRs. It eliminates the need to reconfigure each existing TE LSR in order to establish a full mesh of TE tunnels whenever a new TE LSR is added in the network.

Configuration Example

This example configures an auto-tunnel mesh group and specifies the attributes for the tunnels in the mesh-group.

RP/0/RP0/CPU0:router # configure
RP/0/RP0/CPU0:router(config)# mpls traffic-eng
RP/0/RP0/CPU0:router(config-mpls-te)# auto-tunnel mesh
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh)# tunnel-id min 1000 max 2000
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh)# group 10
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh-group)# attribute-set 10
RP/0/RP0/CPU0:router(config-mpls-te-auto-mesh-group)# destination-list dl-65
RP/0/RP0/CPU0:router(config-mpls-te)# attribute-set auto-mesh 10
RP/0/RP0/CPU0:router(config-mpls-te-attribute-set)# autoroute announce
RP/0/RP0/CPU0:router(config-mpls-te-attribute-set)# auto-bw collect-bw-only
RP/0/RP0/CPU0:router(config)# commit
Verification

Verify the auto-tunnel mesh configuration using the `show mpls traffic-eng auto-tunnem mesh` command.

```
RP/0/RP0/CPU0:router# show mpls traffic-eng auto-tunnel mesh
```

Auto-tunnel Mesh Global Configuration:
- Unused removal timeout: 1h 0m 0s
- Configured tunnel number range: 1000-2000

Auto-tunnel Mesh Groups Summary:
- Mesh Groups count: 1
- Mesh Groups Destinations count: 3
- Mesh Groups Tunnels count:
  - 3 created, 0 up, 3 down, 0 FRR enabled

Mesh Group: 10 (3 Destinations)
- Status: Enabled
- Attribute-set: 10
- Destination-list: dl-65 (Not a prefix-list)
- Recreate timer: Not running

<table>
<thead>
<tr>
<th>Destination</th>
<th>Tunnel ID</th>
<th>State</th>
<th>Unused timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.2</td>
<td>1000</td>
<td>up</td>
<td>Not running</td>
</tr>
<tr>
<td>192.168.0.3</td>
<td>1001</td>
<td>up</td>
<td>Not running</td>
</tr>
<tr>
<td>192.168.0.4</td>
<td>1002</td>
<td>up</td>
<td>Not running</td>
</tr>
</tbody>
</table>

Displayed 3 tunnels, 0 up, 3 down, 0 FRR enabled

Auto-mesh Cumulative Counters:
- Total
  - Created: 3
  - Connected: 0
  - Removed (unused): 0
  - Removed (in use): 0
  - Range exceeded: 0

Configuring an MPLS Traffic Engineering Interarea Tunneling

The MPLS TE Interarea Tunneling feature allows you to establish MPLS TE tunnels that span multiple Interior Gateway Protocol (IGP) areas and levels, removing the restriction that had required the tunnel head-end and tail-end routers both be in the same area. The IGP can be either Intermediate System-to-Intermediate System (IS-IS) or Open Shortest Path First (OSPF). To configure an inter-area tunnel, you specify on the head-end router a loosely routed explicit path for the tunnel label switched path (LSP) that identifies each area border router (ABR) the LSP should traverse using the next-address loose command. The head-end router and the ABRs along the specified explicit path expand the loose hops, each computing the path segment to the next ABR or tunnel destination.

Configuration Example

This example configures an IPv4 explicit path with ABR configured as loose address on the head-end router.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# explicit-path name interarea1
RP/0/RP0/CPU0:router(config-expl-path)# index1 next-address loose ipv4 unicast 172.16.255.129
RP/0/RP0/CPU0:router(config-expl-path)# index 2 next-address loose ipv4 unicast 172.16.255.131
RP/0/RP0/CPU0:router(config)# interface tunnel-te1
RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered Loopback0
RP/0/RP0/CPU0:router(config-if)# destination 172.16.255.2
RP/0/RP0/CPU0:router(config-if)# path-option 10 explicit name interarea1
RP/0/RP0/CPU0:router(config)# commit
```
MPLS-TE Fast Reroute Link and Node Protection

Fast Reroute (FRR) is a mechanism for protecting MPLS TE LSPs from link and node failures by locally repairing the LSPs at the point of failure, allowing data to continue to flow on them while their head-end routers try to establish new end-to-end LSPs to replace them. FRR locally repairs the protected LSPs by rerouting them over backup tunnels that bypass failed links or node.

Backup tunnels that bypass only a single link of the LSP’s path provide link protection. They protect LSPs if a link along their path fails by rerouting the LSP’s traffic to the next hop (bypassing the failed link). These tunnels are referred to as next-hop (NHOP) backup tunnels because they terminate at the LSP’s next hop beyond the point of failure.

The following figure illustrates link protection.

*Figure 7: Link Protection*

FRR provides node protection for LSPs. Backup tunnels that bypass next-hop nodes along LSP paths are called next-next-hop (NNHOP) backup tunnels because they terminate at the node following the next-hop node of the LSP paths, bypassing the next-hop node. They protect LSPs if a node along their path fails by enabling the node upstream of the failure to reroute the LSPs and their traffic around the failed node to the next-next hop. NNHOP backup tunnels also provide protection from link failures, because they bypass the failed link and the node.
The following figure illustrates node protection.

Figure 8: Node Protection

Differentiated Services Traffic Engineering

MPLS Differentiated Services Aware Traffic Engineering (DS-TE) is an extension of the regular MPLS-TE feature. Regular traffic engineering does not provide bandwidth guarantees to different traffic classes. A single bandwidth constraint is used in regular TE that is shared by all traffic. To support various classes of service (CoS), users can configure multiple bandwidth constraints. These bandwidth constraints can be treated differently based on the requirement for the traffic class using that constraint.

Cisco IOS XR software supports two DS-TE modes: pre-standard and IETF. The pre-standard DS-TE mode uses the Cisco proprietary mechanisms for RSVP signaling and IGP advertisements. This DS-TE mode does not interoperate with third-party vendor equipment. Pre-standard DS-TE is enabled only after configuring the sub-pool bandwidth values on MPLS-enabled interfaces. Pre-standard DS-TE mode supports a single bandwidth constraint model a Russian Doll Model (RDM) with two bandwidth pools: global-pool and sub-pool. TE class map is not used with Pre-standard DS-TE mode.

IETF DS-TE mode uses IETF-defined extensions for RSVP and IGP. This mode inter-operates with third-party vendor equipment. IETF mode supports multiple bandwidth constraint models, including RDM and Maximum Allocation Bandwidth Constraint Model (MAM), both with two bandwidth pools. In an IETF DS-TE network, identical bandwidth constraint models must be configured on all nodes. TE class map is used with IETF DS-TE mode and must be configured the same way on all nodes in the network.

The MAM constraint model has the following characteristics:

• Easy to use and intuitive.
• Isolation across class types.
• Simultaneously achieves isolation, bandwidth efficiency, and protection against QoS degradation.

The RDM constraint model has these characteristics:

• Allows greater sharing of bandwidth among different class types.
• Ensures bandwidth efficiency simultaneously and protection against QoS degradation of all class types.
• Specifies that it is used in conjunction with preemption to simultaneously achieve isolation across class-types such that each class-type is guaranteed its share of bandwidth, bandwidth efficiency, and protection against QoS degradation of all class types.

**MPLS-TE Forwarding Adjacency**

MPLS TE forwarding adjacency allows you to handle a TE label-switched path (LSP) tunnel as a link in an Interior Gateway Protocol (IGP) network that is based on the Shortest Path First (SPF) algorithm. Both Intermediate System-to-Intermediate System (IS-IS) and Open Shortest Path First (OSPF) are supported as the IGP. A forwarding adjacency can be created between routers regardless of their location in the network. The routers can be located multiple hops from each other.

As a result, a TE tunnel is advertised as a link in an IGP network with the tunnel's cost associated with it. Routers outside of the TE domain see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network. TE tunnel interfaces are advertised in the IGP network just like any other links. Routers can then use these advertisements in their IGPs to compute the SPF even if they are not the head-end of any TE tunnels.

**Automatic Bandwidth**

Automatic bandwidth allows to dynamically adjust bandwidth reservation based on measured traffic. MPLS-TE automatic bandwidth is configured on individual Label Switched Paths (LSPs) at every head-end router. MPLS-TE automatic bandwidth monitors the traffic rate on a tunnel interface and resizes the bandwidth on the tunnel interface to align it closely with the traffic in the tunnel.

MPLS-TE automatic bandwidth can perform these functions:

- Monitors periodic polling of the tunnel output rate
- Resizes the tunnel bandwidth by adjusting the highest rate observed during a given period

For every traffic-engineered tunnel that is configured for an automatic bandwidth, the average output rate is sampled, based on various configurable parameters. Then, the tunnel bandwidth is readjusted automatically based upon either the largest average output rate that was noticed during a certain interval, or a configured maximum bandwidth value.

While re-optimizing the LSP with the new bandwidth, a new path request is generated. If the new bandwidth is not available, the last good LSP continues to be used. This way, the network experiences no traffic interruptions. If minimum or maximum bandwidth values are configured for a tunnel, the bandwidth, which the automatic bandwidth signals, stays within these values.

The output rate on a tunnel is collected at regular intervals that are configured by using the **application** command in MPLS-TE auto bandwidth interface configuration mode. When the application period timer expires, and when the difference between the measured and the current bandwidth exceeds the adjustment threshold, the tunnel is re-optimized. Then, the bandwidth samples are cleared to record the new largest output rate at the next interval. If a tunnel is shut down, and is later brought again, the adjusted bandwidth is lost and the tunnel is brought back with the initial configured bandwidth. When the tunnel is brought back, the application period is reset.

**MPLS Traffic Engineering Interarea Tunneling**

The MPLS-TE interarea tunneling feature allows you to establish TE tunnels spanning multiple Interior Gateway Protocol (IGP) areas and levels, thereby eliminating the requirement that head-end and tail-end routers reside in a single area.
Interarea support allows the configuration of a TE LSP that spans multiple areas, where its head-end and tail-end label switched routers (LSRs) reside in different IGP areas. Multiarea and Interarea TE are required by the customers running multiple IGP area backbones (primarily for scalability reasons). This lets you limit the amount of flooded information, reduces the SPF duration, and lessens the impact of a link or node failure within an area, particularly with large WAN backbones split in multiple areas.

The following figure shows a typical interarea TE network using OSPF.

**Figure 9: Interarea (OSPF) TE Network Diagram**

The following figure shows a typical interlevel (IS-IS) TE Network.

**Figure 10: Interlevel (IS-IS) TE Network Diagram**

As shown in the figure, R2, R3, R7, and R4 maintain two databases for routing and TE information. For example, R3 has TE topology information related to R2, flooded through Level-1 IS-IS LSPs plus the TE topology information related to R4, R9, and R7, flooded as Level 2 IS-IS Link State PDUs (LSPs) (plus, its own IS-IS LSP).

Loose hop optimization allows the re-optimization of tunnels spanning multiple areas and solves the problem which occurs when an MPLS-TE LSP traverses hops that are not in the LSP's headend's OSPF area and IS-IS level. Interarea MPLS-TE allows you to configure an interarea traffic engineering (TE) label switched path (LSP) by specifying a loose source route of ABRs along the path. Then it is the responsibility of the ABR (having a complete view of both areas) to find a path obeying the TE LSP constraints within the next area to
reach the next hop ABR (as specified on the head-end router). The same operation is performed by the last ABR connected to the tail-end area to reach the tail-end LSR.

You must be aware of these considerations when using loose hop optimization:

- You must specify the router ID of the ABR node (as opposed to a link address on the ABR).
- When multiarea is deployed in a network that contains subareas, you must enable MPLS-TE in the subarea for TE to find a path when loose hop is specified.
- You must specify the reachable explicit path for the interarea tunnel.
Implementing MPLS OAM

MPLS Operations, Administration, and Maintenance (OAM) helps service providers to monitor label-switched paths (LSPs) and quickly isolate MPLS forwarding problems to assist with fault detection and troubleshooting in an MPLS network. This module describes MPLS LSP Ping and Traceroute features which can be used for failure detection and troubleshooting of MPLS networks.

MPLS LSP Ping

The MPLS LSP Ping feature is used to check the connectivity between Ingress LSR and egress LSRS along an LSP. MPLS LSP ping uses MPLS echo request and reply messages, similar to Internet Control Message Protocol (ICMP) echo request and reply messages, to validate an LSP. While ICMP echo request and reply messages validate IP networks, MPLS echo and reply messages validate MPLS networks. The MPLS echo request packet is sent to a target router through the use of the appropriate label stack associated with the LSP to be validated. Use of the label stack causes the packet to be forwarded over the LSP itself. The destination IP address of the MPLS echo request packet is different from the address used to select the label stack. The destination IP address is defined as a 127.x.y.z/8 address and it prevents the IP packet from being IP switched to its destination, if the LSP is broken.

An MPLS echo reply is sent in response to an MPLS echo request. The reply is sent as an IP packet and it is forwarded using IP, MPLS, or a combination of both types of switching. The source address of the MPLS echo reply packet is an address obtained from the router generating the echo reply. The destination address is the source address of the router that originated the MPLS echo request packet. The MPLS echo reply destination port is set to the echo request source port.
The following figure shows MPLS LSP ping echo request and echo reply paths.

Figure 11: MPLS LSP Ping Echo Request and Reply Paths

Configuration Examples

This example shows how to use MPLS LSP ping to test the connectivity of an IPv4 LDP LSP. The destination is specified as a Label Distribution Protocol (LDP) IPv4 prefix and Forwarding Equivalence Class (FEC) type is specified as generic.

```
RP/0/RP0/CPU0:router# ping mpls ipv4 4.4.4.4/32 fec-type generic
```

Sending 5, 100-byte MPLS Echos to 4.4.4.4/32, timeout is 2 seconds, send interval is 0 msec:


Type escape sequence to abort.

```
!!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/3 ms

This example shows how to use MPLS LSP ping to test the connectivity when the destination is specified as a MPLS traffic engineering (TE) tunnel.

```
RP/0/RP0/CPU0:router# ping mpls traffic-eng tunnel-te 4003 source 3.3.3.3
```

Sending 5, 100-byte MPLS Echos to tunnel-te4003, timeout is 2 seconds, send interval is 0 msec:


Type escape sequence to abort.

```
!!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 3/3/4 ms

This example shows how to use the show mpls oam command to display the MPLS OAM information

```
RP/0/RP0/CPU0:router# show mpls oam counters packet
```

Sending 5, 100-byte MPLS Echos to tunnel-te4003, timeout is 2 seconds, send interval is 0 msec:


Type escape sequence to abort.

```
!!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 3/3/4 ms

This example shows how to use the show mpls oam command to display the MPLS OAM information
MPLS LSP Traceroute

The MPLS LSP Traceroute feature is used to isolate the failure point of an LSP. It is used for hop-by-hop fault localization and path tracing. The MPLS LSP Traceroute feature relies on the expiration of the Time to Live (TTL) value of the packet that carries the echo request. When the MPLS echo request message hits a transit node, it checks the TTL value and if it is expired, the packet is passed to the control plane, else the message is forwarded. If the echo message is passed to the control plane, a reply message is generated based on the contents of the request message.

The following figure shows an MPLS LSP traceroute example with an LSP from LSR1 to LSR4.

Figure 12: MPLS LSP Traceroute

![MPLS LSP Traceroute Diagram]

Configuration Examples

This example shows how to use the `traceroute` command to trace to a destination with Forwarding Equivalence Class (FEC) type specified as generic.

RP/0/RP0/CPU0:router# traceroute mpls ipv4 3.3.3.3/32 fec-type generic
Mon Nov 30 17:48:45.585 UTC
Tracing MPLS Label Switched Path to 3.3.3.3/32, timeout is 2 seconds

<table>
<thead>
<tr>
<th>Pkt</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive Counts:</td>
<td></td>
</tr>
<tr>
<td>Good Requests:</td>
<td>0</td>
</tr>
<tr>
<td>Good Replies:</td>
<td>10</td>
</tr>
<tr>
<td>Unknown Pkt Types:</td>
<td>0</td>
</tr>
<tr>
<td>IP header error:</td>
<td>0</td>
</tr>
<tr>
<td>UDP header error:</td>
<td>0</td>
</tr>
<tr>
<td>Runts:</td>
<td>0</td>
</tr>
<tr>
<td>Dropped (Q full):</td>
<td>0</td>
</tr>
<tr>
<td>General error:</td>
<td>0</td>
</tr>
<tr>
<td>Error, no IP:</td>
<td>0</td>
</tr>
<tr>
<td>Error, no memory:</td>
<td>0</td>
</tr>
<tr>
<td>Transmit Counts:</td>
<td></td>
</tr>
<tr>
<td>Good:</td>
<td>10</td>
</tr>
<tr>
<td>Dropped:</td>
<td>0</td>
</tr>
</tbody>
</table>
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'L' - labeled output interface, 'B' - unlabeled output interface,
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
'P' - no rx intf label prot, 'p' - premature termination of LSP,
'R' - transit router, 'I' - unknown upstream index,
'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

0 11.1.1.57 MRU 1500 [Labels: implicit-null Exp: 0]
! 1 11.1.1.58 7 ms23:19