

# Configuring MPLS

# 2

This chapter describes how to configure Multiprotocol Label Switching (MPLS) on your ERX system. MPLS is a hybrid protocol that integrates network layer routing with label switching to provide a layer 3 network with traffic management capability. Traffic engineering enables more effective use of network resources while maintaining high bandwidth and stability.

Topic	Page
Overview	2-1
Features	2-17
Traffic Engineering	2-17
References	2-21
Configuration Tasks	2-22
Explicit Routing	2-48
Configuring Virtual Private Networks	2-52
Configuring IGP and MPLS	2-61
MPLS and Differentiated Services	2-63
Monitoring MPLS	2-64

## Overview

---

MPLS provides traffic-engineering capabilities that make effective use of network resources while maintaining high bandwidth and stability. MPLS enables service providers to provide their customers with the best service available given the provider's resources.

The two basic components of MPLS are label distribution and data mapping. Label distribution is the set of actions MPLS performs to establish and maintain a label-switched path (LSP), also known as an *MPLS tunnel* or *MPLS domain*. Data mapping is the process of getting data packets onto an established LSP.

### *Conventions in This Chapter*

Certain terms used with MPLS, such as the names of messages, are often expressed in the RFCs and other sources either with initial uppercase letters or all uppercase letters. For improved readability, those terms are represented in lowercase in this chapter. Table 2-1 lists the terms and some of their variant spellings.

**Table 2-1** Conventions for MPLS terms

In this chapter	In RFCs and other sources	
ack	Ack	ACK
bundle	Bundle	
hello	Hello	HELLO
initialization	Initialization	INITIALIZATION
keepalive	Keepalive	KEEPALIVE
label mapping	Label Mapping	LABEL_MAPPING
label release	Label Release	LABEL_RELEASE
label request	Label Request	LABEL_REQUEST
label request abort	Label Request Abort	LABEL_REQUEST_ABORT
label withdrawal	Label Withdrawal	LABEL_WITHDRAWAL
message ack	message_Ack	MESSAGE_ACK
message ID	message_ID	MESSAGE_ID
srfresh	Srefresh	
path	Path	PATH
patherr	PathErr	PATHERR
pathtear	PathTear	PATHTEAR
resv	Resv	RESV
resvrr	ResvErr	RESVERR
resvtear	ResvTear	RESVTEAR
targeted hello	Targeted Hello	TARGETED_HELLO

Terms

Table 2-2 defines terms and acronyms that are used in this discussion of MPLS.

**Table 2-2** MPLS terms and acronyms

Term	Definition
Admission control	Accounting mechanism that tracks resource information. Prevents requests from being accepted if sufficient resources are not available.
Constraint-based routing	A mechanism to establish paths based on certain criteria (explicit route, QoS parameters). The standard routing protocols can be enhanced to carry additional information to be used when running the route calculation.
CR-LDP	Constraint-based routed LSP setup using LDP
CR-LSP	Constraint-based routed label-switched path
Explicit routing	A subset of constraint-based routing where the constraint is an explicit route
FEC	Forwarding equivalence class – Group of IP packets forwarded over the same path with the same path attributes applied
Label Distribution Protocol	1) A particular label distribution protocol used for label distribution among the routers in an MPLS domain; represented by the acronym LDP 2) In lowercase—label distribution protocol—a generic term for any of several protocols that distribute labels among the routers in an MPLS domain, including LDP, CR-LDP, and RSVP-TE. This usage is <b>not</b> represented in this text by the acronym, LDP.
LDP	Label Distribution Protocol – a particular protocol used for label distribution among the routers in an MPLS domain This text does <b>not</b> use LDP to refer to the generic class of label distribution protocols.
LSP	Label-switched path, the path traversed by a packet that is routed via MPLS. Some LSPs act as tunnels.
LSP priority level	Indicates the importance of one LSP relative to another LSP. LSPs having higher priorities can preempt LSPs having lower priorities. Priorities range from 0 to 7 in order of <i>decreasing</i> priority.
LSR	Label-switching router, an MPLS node that can forward layer 3 packets based on their labels
MPLS	Multiprotocol Label Switching, set of techniques enabling forwarding of traffic using layer 2 and layer 3 information
MPLS edge node	MPLS node that connects an MPLS domain with a node outside the domain that either does not run MPLS or is in a different domain

**Table 2-2** MPLS terms and acronyms (continued)

Term	Definition
MPLS egress node	MPLS edge node in the role of handling traffic as it leaves an MPLS domain
MPLS ingress node	MPLS edge node in the role of handling traffic as it enters an MPLS domain
MPLS label	Label carried in a packet header that represents a packet's forwarding equivalence class
MPLS node	A router running MPLS. An MPLS node is aware of MPLS control protocols, operates one or more L3 routing protocol(s), and is capable of forwarding packets based on labels. Optionally, an MPLS node can be capable of forwarding native L3 packets.
Provider edge router	PE, an LER at the edge of a service provider core that provides ingress to or egress from a VPN
Provider core router	P, an LSR within a service provider core that carries traffic for a VPN
RSVP	Resource Reservation Protocol; the system does not support RSVP
RSVP-TE	Resource Reservation Protocol enhanced to support MPLS traffic engineering; the system supports RSVP-TE
Traffic engineering	The ability to control the path taken through a network or portion of a network based on a set of traffic parameters (bandwidth, QoS parameters, and so on). Traffic engineering (TE) enables performance optimization of operational networks and their resources.
Tunnel	LSP that is used by an IGP to reach a destination, or an LSP that uses traffic engineering

### *IP Routing*

In conventional IP routing, as a packet traverses from one router to the next through a network, each router analyzes the packet's header and performs a network layer routing table lookup to choose the next hop for the packet. In conventional IP forwarding, the router looks for the address in its forwarding table with the longest match (best match) for the packet's destination address. All packets forwarded to this longest match are considered to be in the same forwarding equivalence class (FEC).

### *Label Switching*

MPLS is not a routing protocol; it works with layer 3 routing protocols (IS-IS, OSPF, BGP) to integrate network layer routing with label switching to provide a layer 3 network with traffic management capability. An MPLS FEC consists of a set of layer 3 packets that are all

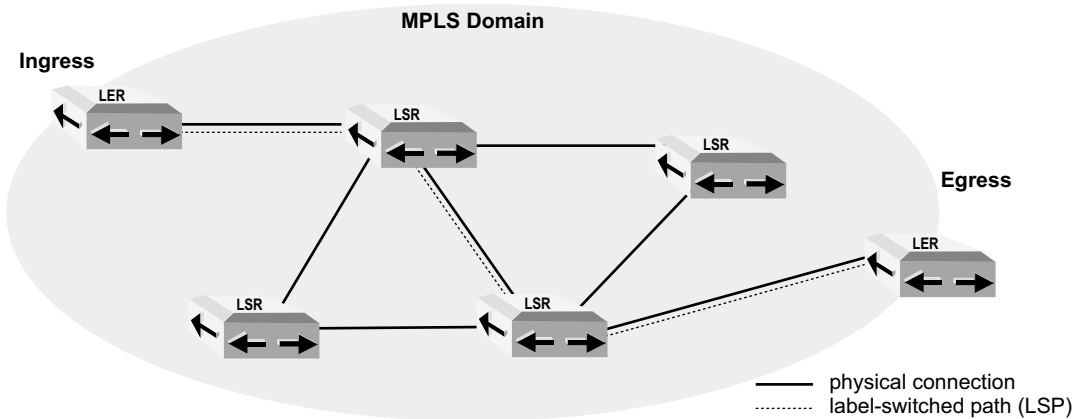
forwarded in the same manner by a given label-switching router (LSR). For example, all packets received on a particular interface might be assigned to a FEC. MPLS assigns each layer 3 packet to a FEC only at the LSR that serves as the ingress node to the MPLS domain. A label distribution protocol binds a label to the FEC. Each LSR uses the label distribution protocol to signal its forwarding peers and distribute its labels to establish an LSP. The label distribution protocol enables negotiation with the downstream LSRs to determine what labels are used on the LSP and how they are employed.

Labels represent the FEC along the LSP from the ingress node to the egress node. The FEC label is prepended to the packet when the packet is forwarded to the next hop. Each label is valid only between a pair of LSRs. A downstream LSR reached by a packet uses the label—instead of information from the layer 3 packet header—as an index into a table that contains both the next hop and a different label to prepend to the packet before forwarding. This table is a modified forwarding table, usually referred to as a label information base (LIB).

The LSR that serves as the egress MPLS node uses the FEC label as an index into a table that has the information necessary to forward the packet from the MPLS domain. The forwarding actions at the egress LSR can be any of the following:

- Pull a second label from the packet and forward the packet based on that label.
- Prepend a new label and send the packet into another MPLS domain.
- Do a route lookup to route the packet to its next hop.

Figure 2-1 shows a simple MPLS domain, consisting of multiple LSRs. The LSRs serving as ingress and egress nodes are also referred to as *label edge routers* (LERs). The ingress router is sometimes referred to as the *tunnel head end*, or the *head-end* router. The egress router is sometimes referred to as the *tunnel tail end*, or the *tail-end* router. LSPs are *unidirectional*, carrying traffic only in the downstream direction from the ingress node to the egress node.



**Figure 2-1** Simple MPLS domain

### Label-Switching Routers

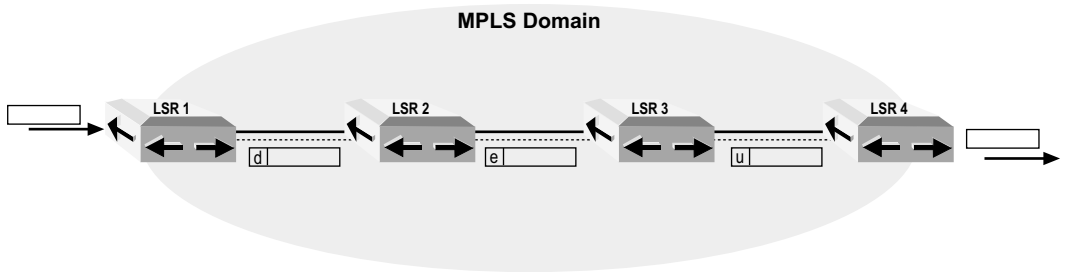
Each LSR, also known as an MPLS node, must have the following:

- At least one layer 3 routing protocol
- A label distribution protocol
- The ability to forward packets based on their labels

The system can use BGP, IS-IS, or OSPF as its layer 3 routing protocol, and LDP, CR-LDP, or RSVP-TE as its label distribution protocol.

MPLS can label packets by using the existing layer 2 header or an encapsulation header that carries the MPLS label. During LSP negotiation, the LSRs in an MPLS domain agree on a labelling method. Labels have only local meaning; that is, meaning for two LSR peers. Each pair of LSRs—consisting of a label originator and a label acceptor—must use a label distribution protocol to agree on the label-to-FEC binding.

Because of the local label assignment, packet labels typically change at each segment in the LSP path, as shown in Figure 2-2. The ingress node, LSR 1, receives an unlabeled data packet and prepends label *d* to the packet. LSR 2 receives the packet, removes label *d* and uses it as an index in its forwarding table to find the next label. LSR 2 prepends label *e* to the packet. LSR 3 does the same thing, removing label *e* and prepending label *u*. Finally, the egress node, LSR 4, removes label *u* and determines where to forward the packet outside the MPLS domain.



**Figure 2-2** Label switching

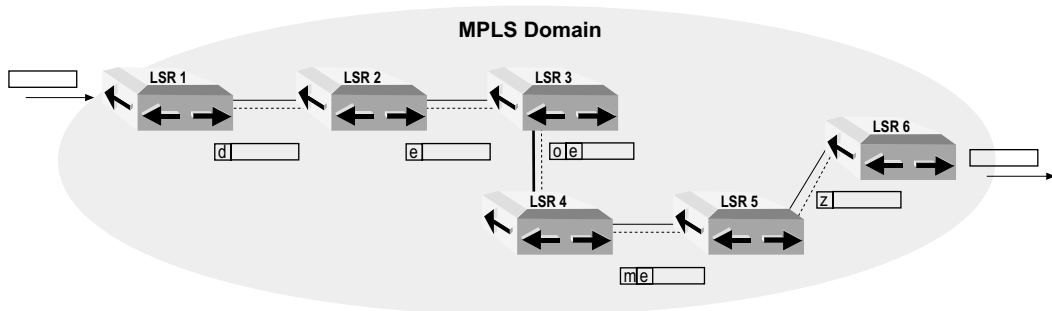
Any packet can carry multiple labels. The labels are stacked in a last-in-first-out order. Each LSR forwards packets based on the outermost (top) label in the stack. An LSR *pushes* a label onto the stack when it prepends the label to a packet header. It *pops* the label when it pulls the label off the stack and compares it with the forwarding table. On determining the label for the next segment of the LSP, the LSR pushes the new label on the stack. A label swap consists of a pop, lookup, and push.

When the egress router, such as LSR 4 in Figure 2-2, receives a packet, it performs two lookups: it looks up the label and determines that the label must be popped, then it does a layer 3 lookup to determine where to forward the packet. Some MPLS implementations use a penultimate hop pop (PHP) to reduce the number of lookups performed by the LER. In PHP, the LER requests its upstream neighbor (the penultimate hop) to pop the final label and send just the IP packet to the LER. The LER then performs only the layer 3 lookup. The request to perform PHP is signaled by the LER by sending an implicit null label in the label mapping message the LER sends to its upstream neighbor. The implicit null label never appears in the encapsulation.

Our MPLS implementation never requests that an upstream neighbor should perform a PHP. However, if an ERX system receives a PHP request from a downstream neighbor, the ERX system will perform the PHP.

Figure 2-3 shows an LSP that uses label stacking. The ingress node, LSR 1, receives an unlabeled data packet and prepends label *d* to the packet. LSR 2 receives the packet, removes label *d* and uses it as an index in its forwarding table to find the next label, prepending label *e* to the packet. LSR 3 removes label *e* and prepends label *s* (negotiated with LSR 5) to the packet. LSR 3 pushes label *x* on top of label *s*. LSR 4 pops the top (outermost) label, *x*, and pushes label *r* on top of label *s*. LSR 5 pops label *r*, determines that it must pop label *s*, and pushes label *z* on the empty

stack. Finally, the egress node, LSR 6, removes label *z* and determines where to forward the packet outside the MPLS domain.



**Figure 2-3** Label stacking

The configuration shown in Figure 2-3 is an example of an LSP within an LSP (a tunnel within a tunnel). The first LSP consists of LSR 1, LSR 2, LSR 3, LSR 5, and LSR 6. The second LSP consists of LSR 3, LSR 4, and LSR 5. The two LSPs have different ingress and egress points. LSR 1 and LSR 6 are LERs. Less obviously, LSR 3 and LSR 5 are also LERs, but for the internal LSP.



**Note:** Label stacking is typically employed for LSR peers that are not directly connected. Figure 2-3 is a simplified example to illustrate the concept of label stacking.

## Labels

MPLS uses labels from either the *platform label space* or the *interface label space*. You can define the range available to the system for labels in either space. ATM interfaces can use labels from either label space; Ethernet interfaces use labels only from the platform label space.

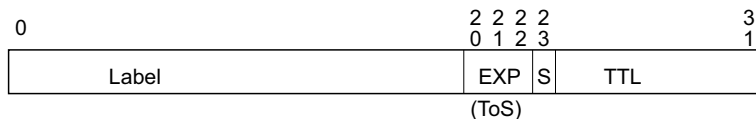
Platform labels are simply integers chosen from the range 16–1048575. Configuring platform labels effectively creates a large single pool of labels that can be shared by the platform—all MPLS interfaces on a given virtual router. In contrast, interface labels enable you to effectively create multiple smaller pools of labels, each used only by a particular interface. When you configure interface labels, you restrict only a given interface to a particular range of labels. Other interfaces in that VR can still use labels from that space unless you restrict them in turn to a different range of interface labels.

In the interface label space, MPLS selects labels from interface resources, either a VPI/VCI combination or only a VCI. You configure a VPI range and a VCI range available to the labels. In the VCI-only case (supported

only for LDP and CR-LDP), the VPI is inherited from an ATM 1483 interface and cannot be configured. When an upstream LSR requests a label, the downstream LSR allocates a VPI/VCI combination for use as a label between these two peers. Allocating labels on a per interface basis is necessary because the VPI/VCI ranges are limited. This enables you to use the same label on different interfaces without conflict.

When you use the platform label space, the MPLS ingress node places labels in *shim headers* in front of the IP packets, between the data link layer header and the network layer header. The shim header includes the following bits (Figure 2-4):

- Label bits – Twenty bits
- EXP bits – Three bits for class of service information; these bits are variously called the experimental bits, class of service (CoS) bits, or type of service (ToS) bits. The EXP bits are mapped from the IP packet at the ingress node and are mapped back into the IP packet at the egress node.
- S bit – One bit to indicate whether the label is on the bottom of the label stack.
- TTL bits – Eight bits for a time-to-live indicator. The TTL bits are mapped from the IP packet at the ingress node. The TTL bits in the shim header are decremented at each hop, arriving at the egress node with the correct count. The bits are mapped back into the IP packet at the egress node.



**Figure 2-4** Shim header

If you configure an MPLS interface to use the interface label space, the data link layer addresses are used as labels, so there is no need to place them within a shim header. The node determines the next-hop label (address) and places it in the ATM address field. As the data travels along the LSP, the LSRs examine only the label in the address field. The shim header is used only to carry the TTL bits to the egress. The ingress node learns the total hop count from signalling and then uses that count to decrement the TTL to the correct final value. The TTL is then carried in the shim header to the egress node without modification, arriving with the correct count.

### *Label Distribution Methodology*

Our implementation of MPLS supports the following methods of label distribution:

- Downstream-on-demand, ordered control
- Downstream-unsolicited, independent control

*Downstream-on-demand* means that MPLS devices do not signal a FEC-to-label binding until requested to do so by an upstream device. Upstream is the direction toward a packet's source; the ingress node in an MPLS domain is the farthest possible upstream node. Downstream is the direction toward a packet's destination; the egress node in an MPLS domain is the farthest possible downstream node.

Downstream-on-demand conserves labels in that they are not bound until they are needed and the LSR receives only label mappings from a neighbor that is the next hop to a destination; it is used when adjacent peers are configured to use the interface label space.

*Ordered control* means that the LSR making the initial request for label negotiation does not receive a response from its peer until the request has been propagated to the packet's destination, or to the egress node of the MPLS domain. This *endpoint* must respond to its upstream peer with a label (label binding, also known as label mapping); that peer in turn responds to its upstream peer with a label, and so on until the initiator receives a label. In this manner the entire LSP is established before MPLS begins to map data onto the LSP, preventing inappropriate (early) data mapping from occurring on the first LSR in the path.

In Figure 2-5, LSR A sends a request for label negotiation to LSR C. Before LSR C responds, it sends its own request to LSR D. LSR D in turn makes a request for a label to LSR F. When LSR F returns an acceptable label to LSR D, that label is for use only between LSRs D and F. LSR D sends a label back to LSR C that this pair of LSRs will use. Finally, LSR C sends back to LSR A the label that they will use. This completes the establishment of the LSP.

*Downstream-unsolicited* means that MPLS devices do not wait for a request from an upstream device before signalling FEC-to-label bindings. As soon as the LSR learns a route, it sends a binding for that route to all peer LSRs, both upstream and downstream. Downstream-unsolicited does not conserve labels, because an LSR receives label mappings from neighbors that might not be the next hop for the destination; it is used by LDP when adjacent peers are configured to use the platform label space.

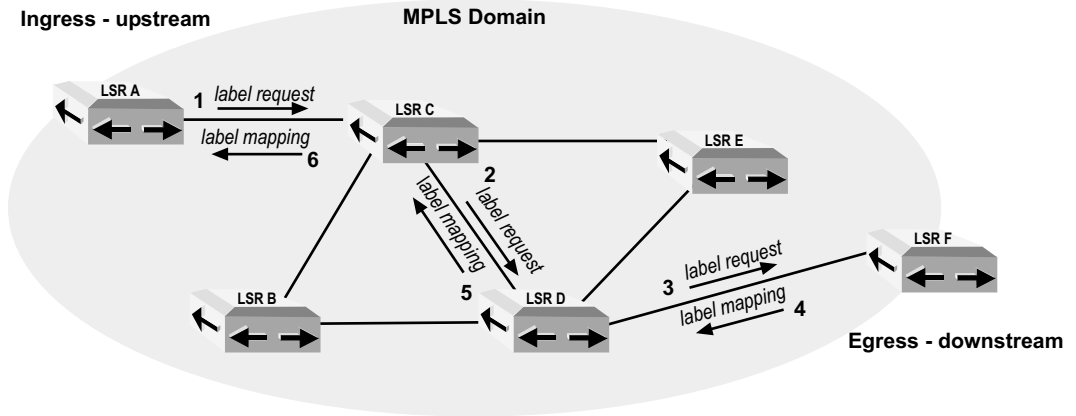


Figure 2-5 LSP creation, downstream-on-demand, ordered control

*Independent control* means that the LSR sending the label acts independently of its downstream peer. It does not wait for a label from the downstream LSR before it sends a label to peers.

In Figure 2-6, LSR D learns a route to some prefix. LSR D immediately maps a label for this destination and sends the label to its peers, LSR B, LSR C, LSR E, and LSR F. In the topology-driven network, the LSPs are created automatically with each peer LSR.

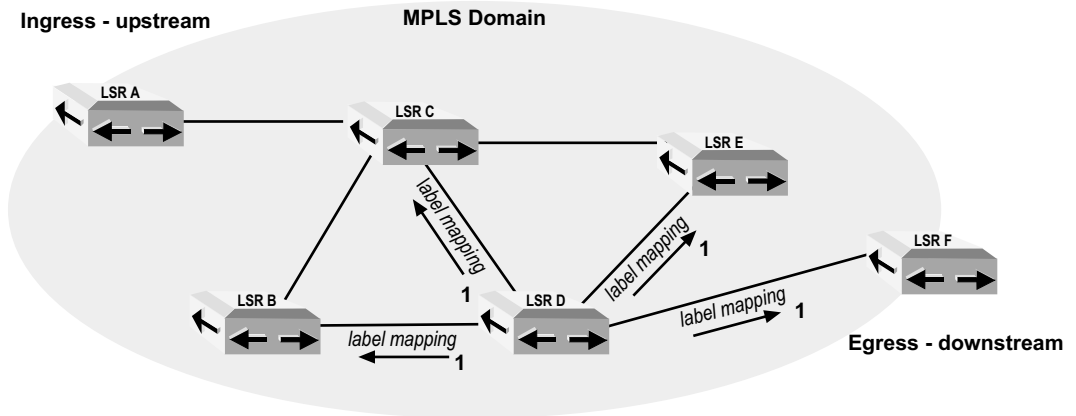


Figure 2-6 LSP creation, downstream-unsolicited, independent control

### Mapping Data

To use enhanced SPF, you must issue the **tunnel mpls autoroute announce** command and specify either BGP, IS-IS, or OSPF as the IGP. When the LSP is established, the ingress LSR announces the LSP endpoint to the IGP. This is also referred to as *registering* the LSP. The

IGP then recalculates the shortest path for all routes destined for or beyond that endpoint. You can choose to register endpoints with both IS-IS and OSPF. The following is an example registration command:

```
host1(config-if)#tunnel mpls autoroute announce isis
```

If you do not use the `autoroute announce` feature, then you must use another method for mapping data onto the tunnel as the next hop. One such mechanism is to configure static routes. You can determine which routes (routes destined for which subnets) should be directed through the LSP and issue the appropriate **ip address next-hop** commands, as shown in the following example:

```
host1(config-if)#ip route 10.15.21.16 tunnel mpls:1
```

You cannot create any static routes until the tunnel interface has been created. However, the tunnel does not have to be active before you create the static routes.

### *Tunnel Endpoints*

You can use any of the following to create an MPLS tunnel:

- A single static endpoint
- Multiple static endpoints
- Dynamic endpoints learned from routing protocols, such as IS-IS and OSPF

You must use tunnel profiles when configuring dynamic endpoints or multiple static endpoints.

### *Label Distribution Protocols*

Label distribution protocols create and maintain the label-to-FEC bindings along an LSP from MPLS domain ingress to MPLS domain egress. A label distribution protocol is a set of procedures by which one LSR informs a peer LSR of the meaning of the labels used to forward traffic between them. It enables each peer to learn about the other peer's label mappings. The label distribution protocol provides the information MPLS uses to create the forwarding tables in each LSR in the MPLS domain.



**Note:** *Label distribution protocols are sometimes referred to as signaling protocols. However, label distribution is a more accurate description of their function and is preferred in this text.*

The following protocols are currently used for label distribution:

- LDP – Label Distribution Protocol
- CR-LDP – Constraint-based Routed Label Distribution Protocol
- RSVP-TE – Resource Reservation Protocol with traffic-engineering extensions that enable label binding and explicit route capability



**Note:** To reduce confusion, this text uses the lower-case term, label distribution protocol, to refer to the generic class of protocols. The acronym, LDP, refers only to the particular protocol named Label Distribution Protocol.

LDP has no traffic-engineering capability and supports only best-effort LSPs. LDP supports topology-driven MPLS networks in best-effort, hop-by-hop implementations. CR-LDP and RSVP-TE are used primarily for MPLS applications that require traffic-engineering (TE) or quality of service (QoS) capabilities, but they also support best-effort LSPs. CR-LDP only defines traffic-engineering enhancements to LDP and relies on the basic functionality provided by conventional LDP.

#### LDP Messages and Sessions

LDP creates reliable sessions by running over TCP. You do not have to explicitly configure LDP peers, because each LSR actively discovers all other LSRs to which it is directly connected. LDP is a *hard-state* protocol, meaning that once the LSP is established, it is assumed to remain in place until it has been explicitly torn down. This is in contrast to RSVP-TE, which is a *soft-state* protocol. See *RSVP-TE Messages and Sessions* on p 2-15.

LDP uses many messages to create LSPs, classified in the following four types:

- Discovery – to identify other LSRs
- Adjacency – to create, maintain, and end sessions between LSRs
- Label advertisement – to request, map, withdraw, and release labels
- Notification – to provide advisory and error information

Unlike the other LDP messages, the discovery process runs over UDP. Each LSR periodically broadcasts a hello message to the well-known UDP port, 646. Each LSR listens on this port for hello messages from other LSRs. In this manner, each LSR learns about all other LSRs to which it is directly connected, creating hello adjacencies. When an LSR learns about another LSR, it establishes a TCP connection to the peer on well-known TCP port 646 and creates an LDP session on top of the TCP connection.

LDP can also discover peers that are not directly connected if you provide the LSR with the IP address of one or more peers via an access list. The LSR sends targeted hello messages to UDP port 646 on each remote peer. If the targeted peer responds with a targeted hello to the initiator, session establishment can proceed.

The LDP peers exchange session initialization messages that include timer values and label ranges. An LSR responds with a keepalive message if the values in the initialization message are acceptable. If any value is not acceptable, the LSR responds instead with an error notification message, terminating the session. Once a session is established, LDP peers exchange keepalive messages that verify continued functioning of the LSR. Failure to receive an expected keepalive message causes an LSR to terminate the LDP session.

You can configure MPLS to use LDP in either a user-driven, statically configured network or a topology-driven network. In the user-driven network, you perform global, interface, and tunnel (optionally, tunnel profile) configuration tasks. In the topology-driven network, you perform only global configuration tasks; LDP automatically creates LSPs to peers as it learns new IGP routes. Topology-driven networks are implemented for best-effort, hop-by-hop routing.

Label mapping and distribution proceed according to one of the following methods:

- Downstream-on-demand, ordered control is used if you implement a user-driven network, or if you implement a topology-driven network and configure peer LSRs to use the interface label space.
- Downstream-unsolicited, independent control is used if you implement a topology-driven network and configure peer LSRs to use the platform label space.

With downstream-on-demand, ordered control, the upstream LSR sends out a label request to its downstream peer. Each peer in the path to the endpoint in turn sends a label request message. The endpoint returns a label mapping message (binding the label to the FEC) to its upstream peer, and so on, until the initiating LSR receives a label mapping message from its downstream peer. The upstream peer does not send out a label mapping message upstream until it receives one from downstream.

With downstream-unsolicited, independent control, an LSR creates a label binding whenever it learns a new IGP route; the LSR sends a label mapping message immediately to all of its peer LSRs—upstream and downstream—without having received a label request message from any peer. The LSR sends the label mapping message regardless of whether it has received a label mapping message from a downstream LSR. This is

the label distribution method employed in a topology-driven MPLS network.

In the topology-driven network, both downstream-unsolicited and downstream-on-demand modes lead to the development of a fully meshed MPLS domain, enabling MPLS to respond quickly when a connection goes down because alternate paths have already been established.

An LSR that requested a label can prevent the request from being answered by issuing a label request abort message. This might happen if the next hop changes for the FEC for which the label was requested. Similarly, a downstream LSR can send a label withdrawal message to recall a label that it previously mapped. If an LSR that has received a label mapping subsequently determines that it no longer needs that label, it can send a label release message that frees the label for use.

#### RSVP-TE Messages and Sessions

RSVP is described in RFC 2205. Multiple RFCs enable extensions to RSVP for traffic engineering. The system supports the extended version of RSVP, referred to as RSVP-TE.

RSVP-TE is “unreliable” because it does not use TCP to exchange messages. In contrast to LDP—a hard-state protocol—RSVP-TE is a *soft-state* protocol, meaning that much of the session information is embedded in a state machine on each LSR. The state machine must be refreshed periodically to avoid session termination. LSRs send path messages to downstream peers to create and refresh local path states. LSRs send resv messages to upstream peers in response to path messages to create and refresh local resv states. A session is ended if the state machine is not refreshed within the RSVP tunnel timeout period, which is determined as follows:

$$\text{RSVP tunnel timeout period in seconds} = \left\{ [(\text{cleanup timeout factor} + 0.5) \times 1.5] \times \left( \frac{\text{refresh period}}{1000} \right) \right\}$$

For example, for the default values,

$$\text{RSVP tunnel timeout period in seconds} = \left\{ [(3 + 0.5) \times 1.5] \times \left( \frac{30,000}{1000} \right) \right\} = 157.5$$

RSVP-TE messages carry *objects* consisting of type-length-values (TLVs). The *label request object* instructs the endpoint LSR to return an resv message to establish the LSP. The resv message contains the *label object*, the label used for the FEC. Both the path and resv messages carry the *record route object*, which records the route traversed by the message.

An upstream LSR sends a `pathtear` message when its path state times out as a result of not being refreshed. The `pathtear` message removes the path and `resv` states in each LSR as it proceeds downstream. Downstream LSRs similarly send the `resvtear` message when their `resv` state times out to remove the `resv` states in upstream LSRs.

If a downstream LSR determines that it received an erroneous path message, it sends a `patherr` message to the sender. If a reservation (label) request fails, the request initiator sends a `resvrr` message to the downstream LSRs. Both of these messages are advisory and do not alter path or `resv` state.

### RSVP-TE State Refresh and Reliability

RSVP-TE employs refresh messages to synchronize state between neighbors and to recover from lost RSVP-TE messages of other types. RSVP-TE by default does not provide for reliable transmission. Each node is responsible for the transmission of RSVP-TE messages to its neighbors and relies on periodic state refreshes to recover from lost messages.

RSVP-TE refresh reduction provides a means to increase reliability while reducing message handling and synchronization overhead. Issuing the **`mpls rsvp refresh-reduction`** command enables the following features:

- The message ID object is included in RSVP-TE messages to provide a unique message ID on a per-hop basis. In refresh messages, it indicates to the receiving node that the message is a refresh message, eliminating the need for the node to completely analyze the message and thus reducing the processing overhead at the receiver.
- RSVP-TE uses a message acknowledgement mechanism to support reliable message delivery on a per-hop basis. Messages that include the message ID object also include a request for acknowledgement. Upon receipt, the receiving node returns a message ack object, enabling the sending node to determine whether a message was lost and triggering a retransmission as necessary.
- Summary refresh (`srefresh`) messages refresh the state previously advertised in path or `resv` messages that included message ID objects. The `srefresh` message carries the unique message ID as state identifier, eliminating the need to send whole refresh messages and reducing the overhead needed to maintain RSVP-TE state synchronization. This method maintains RSVP-TE's ability to indicate when state is lost and to adjust to changes in routing. The `srefresh` message can carry message IDs for multiple RSVP-TE sessions.

Issuing the **mpls rsvp message-bundling** command enables RSVP-TE to use bundle messages, each of which includes multiple standard RSVP-TE messages, to reduce the overall message processing overhead.

## Features

---

MPLS currently supports the features listed in the following table. See the current *ERX Release Notes* for information on modules supporting MPLS. MPLS on POS interfaces is supported only over PPP. No other encapsulation on the POS interface is supported for MPLS

Admission control on ATM AAL5, ATM1483, and Ethernet interfaces	LER functionality
Best-effort explicit routing	LSP failure recovery
Best-effort hop-by-hop routing	LSR functionality
Constraint-based explicit routing	Traffic engineering
Enhanced data mapping	Virtual private networks
Label stacking	VLANs

## Traffic Engineering

---

MPLS traffic engineering (TE) is the ability to establish all LSPs according to particular criteria (*constraints*) in order to meet specific traffic requirements rather than relying on the path chosen by the conventional IGP. The constraint-based IGP examines the available network resources and calculates the shortest path for a particular tunnel that has the resources required by that tunnel. TE enables you to make the best use of your network resources by reducing overuse and underuse of certain links.

*Constraint-based routing* (CR) makes TE possible by considering resource requirements and resource availability rather than merely the shortest path calculations. Constraints are determined at the edge of the network and include criteria such as required values for bandwidth or required explicit paths. You can use either CR-LDP or RSVP-TE as the label distribution protocol for traffic engineering. The IGP propagates resource information throughout its network. Both RSVP-TE and CR-LDP employ downstream-on-demand, ordered control for label mapping and distribution.

Explicit routing specifies a list or group of nodes (hops) that must be used in setting up the tunnels. CR explicit paths can be *strict* or *loose*. Strict paths specify an exact physical path, including every physical node.

Loose paths include hops that have local flexibility; the hop can be a traditional interface, an autonomous system, or an LSP.

### *LSP Backup*

You can configure multiple LSPs to the same destination. By configuring different tunnel metrics for these LSPs, you can force a ranking or priority of use for the LSPs. In this scenario, all the configured LSPs are up and active. If the LSP in use develops problems and goes down, traffic is diverted to the LSP having the next best metric.

### *Path Option*

You can configure multiple paths for an LSP with the **tunnel mpls path-option** command. Each path option has an identifying number; the lower the number the higher the preference for that path option. In this scenario, only a single LSP is up and active at a time. If the path option currently in use by an LSP goes down, MPLS tries to reroute the tunnel using the path option with the next highest preference. In certain circumstances—for example, when a tunnel is preempted by another—MPLS first attempts to reroute the tunnel with the current path option.

### *Reoptimization*

You can use the traffic-engineering reoptimization capability to ensure that the best path is being used. Suppose the current path goes down and MPLS switches to an alternate path that is not as good as the failed path. You can have MPLS periodically check—according to a specified schedule—for a path better than the alternate by configuring the reoptimization timer. For example, you might configure MPLS to check for a better path every 10 seconds; if it finds a better path, it switches.

On the other hand, you might be concerned about route flapping. If a path goes down and then comes back up, perhaps it will continue to do so. In this case, you might not ever want to go back to a path that goes down. To accomplish this, you can configure reoptimization to never occur.

When you do not want the initial path to change—that is, when you want to pin the route—you can disable reoptimization globally by setting the timer to 0. Alternatively, you can disable reoptimization on a per-tunnel basis by using the **lockdown** option with the **tunnel mpls path-option** command. LSP paths are always pinned until the next reoptimization.

Finally, you can manually force an immediate reoptimization. See *Global Configuration Tasks* later in this chapter for information on configuring reoptimization.

### *Tracking Resources for Traffic Engineering*

MPLS traffic engineering uses *admission control* to keep track of resource information. Admission control has an accounting feature that ensures that requests are not accepted when the system does not have sufficient resources to accommodate them.



**Note:** *Bandwidth accounting is the only feature of admission control that is currently supported.*

Currently, bandwidth (BW) and bandwidth-related information are the only resources tracked and used for traffic engineering. Admission control determines whether a setup request can be honored for an MPLS LSP with traffic parameters.

Admission control provides bandwidth information to the IGP protocols, ISIS and OSPF. As new LSPs are created, the available bandwidth decreases. The IGPs can subsequently advertise this information and use it for SPF calculations to determine paths that satisfy the traffic requirements. You can configure readvertisement to occur periodically or when the change crosses some threshold.

Admission control is not responsible for resource allocation, traffic monitoring, traffic queuing, or traffic policing. That is, when admission control admits a request for an LSP, it does not guarantee that the LSP gets the required resources. Admission control currently uses an algorithm based on simple average bandwidth to account for system resources.

### Starting Admission Control

Admission control operates on a system-wide basis rather than a per-virtual-router basis. Admission control of resources begins when either of the following occurs:

- You configure resource-related information on an interface, including bandwidth (either total bandwidth or MPLS reservable bandwidth), flooding frequency, flooding threshold, administrative weight, or attribute flags.
- MPLS begins to use admission control services; for example, by attempting to set up a constraint-based LSP.

### Admission Control Interface Table

Configuring bandwidth on an interface creates an entry for the interface in the admission control interface table. Each entry in the table stores the following information per interface:

- Maximum (physical or line-rate) bandwidth
- Maximum reservable bandwidth
- The following information per IP class (currently a single, default class)
  - > Total available (unreserved) bandwidth
  - > Available bandwidth at each MPLS priority level
- Resource flooding threshold and period

The resource flooding threshold and period together control the flooding of the resource information by the IGP protocols, ISIS and OSPF.

### Configuring Traffic-Engineering Resources

You can configure the following resource-related information on an MPLS interface (at either the major interface or subinterface level):

- Bandwidth – total bandwidth that can be reserved on the interface
- Flooding thresholds – sets of absolute percentages of total reservable bandwidth that trigger the new bandwidth value to be flooded throughout the network; flooding is triggered when bandwidth increases past any up threshold value or decreases past any down threshold value
- Flooding frequency – periodicity with which the bandwidth value is flooded, apart from any flooding due to value changes
- Administrative weight – weight assigned to the interface that supersedes any assigned by the IGP
- Attribute flags – 32-bit value that assigns the interface to a resource class and enables a tunnel to discriminate among interfaces by matching against tunnel affinity bits

### Monitoring Resources

See *Monitoring MPLS* in this chapter for information on displaying information related to traffic-engineering resources.

## LSP Preemption

You can develop a preemption strategy whereby a new LSP can claim resources from an existing LSP. Each tunnel can be configured with a *setup* priority and a *hold* priority. Priority levels range from 0 (highest priority) to 7 (lowest priority).

If TE admission control determines that there are insufficient resources to accept a request to set up a new LSP, the setup priority is evaluated against the hold priority of existing LSPs. An LSP with a hold priority lower than the setup priority of the new LSP can be preempted. The existing LSP is terminated to make room (free resources) for the new LSP. You must assign priorities according to network policies to prevent resource poaching and LSP thrashing.

## References

---

For more information about the MPLS protocol, consult the following resources:

- A Framework for Multiprotocol Label Switching – draft-ietf-mpls-framework-06.txt (June 2001 expiration)
- *ERX Release Notes, Appendix A, System Maximums* – refer to the Release Notes corresponding to your software release for information on maximum values.
- RFC 2205 – Resource ReSerVation Protocol (RSVP) -- Version 1, Functional Specification (September 1997)
- RFC 2209 – Resource ReSerVation Protocol (RSVP) -- Version 1, Message Processing Rules (September 1997)
- RFC 2210 – The Use of RSVP with IETF Integrated Services (September 1997)
- RFC 2211 – Specification of the Controlled-Load Network Element Service (September 1997)
- RFC 2547 – BGP/MPLS VPNs (March 1999)
- RFC 2685 – Virtual Private Networks Identifier (September 1999)
- RFC 2702 – Requirements for Traffic Engineering over MPLS (September 1999)
- RFC 2747 – RSVP Cryptographic Authentication (January 2000)
- RFC 2961 – RSVP Refresh Overhead Reduction Extensions (April 2001)

- RFC 3031 – Multiprotocol Label Switching Architecture (January 2001)
- RFC 3032 – MPLS Label Stack Encoding (January 2001)
- RFC 3035 – MPLS using LDP and ATM VC Switching (January 2001)
- RFC 3036 – LDP Specification (January 2001)
- RFC 3037 – LDP Applicability (January 2001)
- RFC 3209 – RSVP-TE: Extensions to RSVP for LSP Tunnels (December 2001)
- RFC 3210 – Applicability Statement for Extensions to RSVP for LSP-Tunnels (December 2001)



**Note:** IETF drafts are valid for only 6 months from the date of issuance. They must be considered as works in progress. Please refer to the IETF Web site at <http://www.ietf.org> for the latest drafts.

## Configuration Tasks

---

Configuring MPLS consists of the following sets of tasks:

- Global tasks – Perform these tasks to configure settings common to all MPLS usage on a given LSR
- Interface tasks – Perform these tasks to configure each interface on an LSR that uses MPLS
- Tunnel tasks – Perform these tasks to configure MPLS tunnels (label-switched path)
- Tunnel profile tasks – Perform these tasks to configure a profile that contains settings to be used by multiple MPLS tunnels

Many users find it convenient to configure MPLS by completing the tasks in each category before moving to the next category. However, you do not have to complete the tasks in the listed order.

To implement a topology-driven network (for best-effort, hop-by-hop, LDP implementations), perform only the global configuration tasks.

## Global Configuration Tasks

Complete these tasks to configure a virtual router as an LSR. These commands are performed in Global Configuration mode. The following sequence is arbitrary; you can perform these tasks in any order.

- 1 Enable MPLS on a virtual router.

```
host1(config)#mpls
```

- 2 (Optional) Configure the acceptable range for labels (from the platform label space).

```
host1(config)#mpls label-range 2300 54320
```

- 3 (Optional) Configure the interval at which the bandwidth values are flooded.

```
host1(config)#mpls traffic-eng link-management timers  
periodic-flooding 10
```

- 4 (Optional) Configure reoptimization—how often MPLS checks for better paths for the tunnel.

```
host1(config)#mpls reoptimize timers frequency 180
```

You can also force an immediate check for better paths for all existing LSPs, as shown in the following example:

```
host1#mpls reoptimize
```

- 5 (Optional) Configure lists of peer addresses that targeted hellos are sent to or accepted from.

```
host1(config)#mpls ldp targeted-hello send list boston5  
host1(config)#mpls ldp targeted-hello receive list concord3
```

- 6 (Optional) Configure a value for the TTL field placed in the MPLS header when a label is first assigned to a packet.

```
host1(config)#no mpls ip propagate-ttl local
```

- 7 (Optional) Configure retry timer options globally (to apply to all tunnels) to set up an LSP after a setup failure. You can also configure timers for a specific tunnel; see *Tunnel Configuration Tasks* later in this chapter.

- CR-LDP and RSVP-TE modes

```
host1(config)#mpls lsp no-route retries 100  
host1(config)#mpls lsp no-route retry-time 45  
host1(config)#mpls lsp retries 250  
host1(config)#mpls lsp retry-time 65
```

- Topology-driven mode

```
host1(config)#mpls ldp no-route retries 100
host1(config)#mpls ldp no-route retry-time 45
host1(config)#mpls ldp retries 250
host1(config)#mpls ldp retry-time 65
```

- 8 Configure LDP session values.

```
host1(config)#mpls ldp session retry-time 2
host1(config)#mpls ldp session retries 1800
```

- 9 For topology-driven mode only, configure the LSR to create topology-driven LSPs.

```
host1(config)#mpls topology-driven-lsp
```

- 10 (Optional) For topology-driven mode only, specify access lists that filter incoming labels and determine the peers to which the labels are advertised.

```
host1(config)#mpls topology-driven-lsp ip-interfaces
access-list label_in
```

- 11 (Optional) For topology-driven mode only, specify the LSPs over which IP interfaces can be created.

```
host1(config)#mpls topology-driven-lsp ip-interfaces egress
```

- 12 (Optional) For topology-driven mode only, establish a policy governing the distribution of incoming LDP labels.

```
host1(config)#mpls ldp advertise-labels for boston1
```

- 13 (Optional) For topology-driven mode only, force the reapplication of policies to all LDP LSPs.

```
host1#clear mpls ldp
```

- 14 (Optional) For RSVP-TE only, enable refresh reduction and message bundling.

```
host1(config)#mpls rsvp refresh-reduction
host1(config)#mpls rsvp message-bundling
```

- 15 (Optional) Configure profile settings, changing the values from the implicit default values.

- LDP

```
host1(config)#mpls ldp profile ldp5
host1(config-ldp)#hello hold-time 2
host1(config-ldp)#cr-ldp
```

- **RSVP-TE**

```
host1(config)#mpls rsvp profile rsvp4
host1(config-rsvp)#refresh-period
host1(config-rsvp)#cleanup-timeout-factor
```

When you have completed the global configuration tasks, proceed to the next section, *Interface Configuration Tasks*.

### ***cleanup-timeout-factor***

- Use to specify the number of refresh messages that can be lost before the PATH or RESV state is ended.
- Together with the refresh period, defines the RSVP tunnel timeout period. See *RSVP-TE Messages and Sessions* earlier in this chapter for more information.
- Example

```
host1(config-rsvp)#cleanup-timeout-factor 9
```

- Use the **no** version to restore the default value of 3.

### ***clear mpls ldp***

- Use to force the reapplication of all policies or access lists to all configured topology-driven LDP LSPs or to restart the LSP creation process.
- Use this command when you have modified existing or created new policies or access lists (with the **mpls topology-driven-lsp ip-interfaces** and **mpls ldp advertise-labels** commands) and want them to be applied to LDP LSPs that are already in an up state.
- This command removes and then reestablishes all existing LDP LSPs; this might affect data traffic on an LSP that is in use when you issue the command.

- Example

```
host1#clear mpls ldp
```

- There is no **no** version.

### ***cr-ldp***

- Use to specify that the traffic-engineering extensions to LDP are in effect on this interface. CR-LDP is enabled by default.

- Example

```
host1(config-ldp)#cr-ldp
```

- Use the **no** version to disable CR-LDP in the profile.

### ***hello hold-time***

- Use to specify the period for which a sending LSR maintains a record of hello messages from the receiving LSR without receipt of another hello from that LSR.
- Each LSR peer sends the hold time in its hello messages; peers negotiate to use the minimum of the hold times proposed by the pair of LSRs.

- Specify a number in the range 0–65535, corresponding to the period in seconds, except for two special values:
  - › 0 indicates 15 seconds for link hellos and 45 seconds for targeted hellos
  - › 65535 indicates an infinite hold time
- Example

```
host1(config-ldp)#hello hold-time 55
```
- Use the **no** version to restore the default value, 0.

### ***mpls***

- Use to create, enable, disable, or delete MPLS on a virtual router.
- MPLS does not exist by default and must be created, either explicitly with this command, or implicitly with another **mpls** configuration command. If you create MPLS implicitly, for example by issuing the command **mpls lsp retries 10**, MPLS remains disabled until you enable it, either by issuing the **mpls** command or the **no mpls disable** command.
- Example

```
host1(config)#mpls
```
- Use the **mpls disable** version to stop MPLS on the virtual router while leaving the MPLS global configuration intact.
- Use the **no mpls** version to halt MPLS on the virtual router and delete the MPLS configuration.
- Use the **no mpls disable** version to enable MPLS that was previously disabled or implicitly created on the virtual router.

### ***mpls ip propagate-ttl***

- Use to configure the time-to-live field placed in the MPLS header when a label is first added to an IP packet.
- This command is enabled by default, causing the TTL field to be copied from the IP packet header and enabling the **traceroute** command to show all the hops in the network. This command also propagates the label TTL into the IP header at the tunnel egress.
- This command applies only to the tunnel ingress; there is no need to issue this command on the router where the tunnel egress is located.
- Example

```
host1(config)#mpls ip propagate-ttl
```
- Use the **default** version to revert to the global default, causing the TTL field to be copied from the IP packet header and enabling the **traceroute** command to show all the hops in the network. The **default** version propagates the label TTL into the IP header at the tunnel egress.
- Use the **no** version to specify a fixed TTL value of 255 for the first label of the MPLS header and hide the structure of the MPLS network (all packets) from all traffic, preventing a **traceroute** command from discovering and displaying LSP hops.

You can specify the types of packets to be hidden from **traceroute** by using the following keywords with the **no** version:

- › **forwarded** – **traceroute** cannot show the hops for forwarded packets; this most common application of the command enables you to hide the structure of the MPLS network from your customers:

```
host1(config)#no mpls ip propagate-ttl forwarded
```

- › **local** – **traceroute** cannot show the hops for local packets

```
host1(config)#no mpls ip propagate-ttl local
```

You can subsequently specify these keywords with the affirmative version of the command to stop hiding the packets from **traceroute**.

### ***mpls label-range***

- Use to specify the range for the platform label space; all configured virtual routers can share this range.
- Ethernet interfaces can use only the platform label space; ATM interfaces can use the platform label space or the interface label space.
- Example

```
host1(config)#mpls label-range 2300 54320
```
- Use the **no** version to restore the default values, 16–1048575.

### ***mpls ldp advertise-labels***

- Use to control LDP label distribution to neighbors.
- You can issue the command one or more times to construct a filter that determines whether and where incoming (locally assigned) labels are distributed. If you do not specify a *toAccessList*, the action is taken for all peers.
- When the LSR learns an IGP route and tries to decide whether to advertise a label for the destination to a particular LDP neighbor, it attempts to match the destination against any access lists specified by this command, in the order in which the commands were issued. The first match determines the action taken, and no further matching is attempted for that destination. If no match is found, the label is distributed to all peers, unless you issued the **no** version, which prevents the label from being distributed to any peer.
- Example

```
host1(config)#mpls ldp advertise-labels for net25 to euro3
host1(config)#mpls ldp advertise-labels for boston1
host1(config)#no mpls ldp advertise-labels
```

In this example, suppose the LSR receives a label for destination 10.10.11.12. If net25 specifies 10.10.11.12, then the access list action—permit or deny—is taken with the destination label for the peers specified in euro3. If net25 does not include 10.10.11.12, the LSR attempts to match it against boston1. If 10.10.11.12 is present in that access list, the specified action is taken for all peers. If boston1 does not include the destination, the last command ensures that the label is not advertised to any peer.

- Use the **no** version to specify that no incoming labels are distributed or to negate label distribution according to the specified keywords.

### ***mpls ldp no-route retries***

- Use to specify the number of attempts that will be made to set up an LSP for topology-driven LDP after a failure due to no available route.
- The default value of 0 means the attempts will be made until successful.
- Example

```
host1(config)#mpls ldp no-route retries 3200
```
- Use the **no** version to restore the default value, 0.

### ***mpls ldp no-route retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for topology-driven LDP after a failure due to no available route.
- Example

```
host1(config)#mpls ldp no-route retry-time 45
```
- Use the **no** version to restore the default value, 30.

### ***mpls ldp retries***

- Use to specify the number of attempts that will be made to set up an LSP for topology-driven LDP after a failure *other* than one due to no available route.
- The default value of 0 means the attempts will be made until successful.
- Example

```
host1(config)#mpls ldp retries 1275
```
- Use the **no** version to restore the default value, 0.

### ***mpls ldp retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for topology-driven LDP after a failure *other* than one due to no available route.
- Example

```
host1(config)#mpls ldp retry-time 15
```
- Use the **no** version to restore the default value, 30.

### ***mpls ldp session retries***

- Use to specify the number of attempts that will be made to set up an MPLS LDP session.
- The default value of 0 means the attempts will be made until successful.
- Example

```
host1(config)#mpls ldp session retries 1800
```
- Use the **no** version to restore the default value, 0.

### ***mpls ldp session retry-time***

- Use to specify the interval in seconds between attempts to set up an MPLS LDP session.
- Example  

```
host1(config)#mpls ldp session retry-time 2
```
- Use the **no** version to restore the default value, 30.

### ***mpls ldp targeted-hello receive list***

- Use to configure the list of peer addresses from which MPLS accepts targeted hello messages.
- You can specify one or more access lists or IP addresses as members of the list.
- Example  

```
host1(config)#mpls ldp targeted-hello receive list concord3
```
- Use the **no** version to remove the list of peer addresses.

### ***mpls ldp targeted-hello send list***

- Use to configure the list of peer addresses to which MPLS sends targeted hello messages.
- You can specify one or more access lists or IP addresses as members of the list.
- Example  

```
host1(config)#mpls ldp targeted-hello send list boston5
```
- Use the **no** version to remove the list of peer addresses.

### ***mpls lsp no-route retries***

- Use to specify the number of attempts that will be made to set up an LSP for CR-LDP and RSVP-TE after a failure due to no available route.
- The default value of 0 means the attempts will be made until successful.
- Example  

```
host1(config)#mpls lsp no-route retries 3200
```
- Use the **no** version to restore the default value, 0.

### ***mpls lsp no-route retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for CR-LDP and RSVP-TE after a failure due to no available route.
- Example  

```
host1(config)#mpls lsp no-route retry-time 45
```
- Use the **no** version to restore the default value, 5.

### ***mpls lsp retries***

- Use to specify the number of attempts that will be made to set up an LSP for CR-LDP and RSVP-TE after a failure *other* than one due to no available route.
- The default value of 0 means the attempts will be made until successful.
- Example

```
host1(config)#mpls lsp retries 1275
```

- Use the **no** version to restore the default value, 0.

### ***mpls lsp retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for CR-LDP and RSVP-TE after a failure *other* than one due to no available route.
- Example

```
host1(config)#mpls lsp retry-time 15
```

- Use the **no** version to restore the default value, 5.

### ***mpls profile***

- Use to create or modify a configuration profile for the specified label distribution protocol—either LDP or RSVP-TE. Places the CLI in either LDP Configuration mode or RSVP Configuration mode.
- When you issue this command from Global Configuration mode, the **interface** keyword is currently optional in the affirmative version of the command, mandatory in the **no** version. In a future release it may become mandatory in both versions. We recommend that you use this keyword for both versions.
- If you do not specify a profile name, the factory default profile is assumed. If you specify a profile not previously configured, it is created with the factory default settings.
- Any change to a profile affects all interfaces that use the profile; changes are effected the next time the interface comes up.
- The following values are defined for the implicit default profiles:

#### RSVP

- › refresh period – 30,000 ms (30 seconds)
- › timeout factor – 3

#### LDP

- › cr-ldp admin state – enabled

- Example

```
host1(config)#mpls rsvp interface profile rsvp1
```

- Use the **no** version to delete the profile.

### ***mpls reoptimize***

- Use to perform an immediate check for better paths for all existing LSPs.
- Example

```
host1#mpls reoptimize
```
- There is no **no** version.

### ***mpls reoptimize timers frequency***

- Use to specify the frequency at which existing LSPs are checked for better paths.
- A value of zero means no reoptimization is performed.
- Example

```
host1(config)#mpls reoptimize timers frequency 86400
```
- Use the **no** version to restore the default value, 3600 seconds.

### ***mpls rsvp message-bundling***

- Use to enable RSVP-TE to send bundle messages, each of which includes multiple standard RSVP-TE messages, to reduce the overall message processing overhead.
- Example

```
host1(config)#mpls rsvp message-bundling
```
- Use the **no** version to disable RSVP-TE message bundling.

### ***mpls rsvp refresh-reduction***

- Use to enable RSVP-TE summary refresh and reliability features, including the message ID object, the message ack object, and summary refresh messages.
- Example

```
host1(config)#mpls rsvp refresh-reduction
```
- Use the **no** version to disable summary refresh and reliability.

### ***mpls topology-driven-lsp***

- Use to implement a topology-driven network by specifying that the LSR automatically creates LSPs when it learns a new IGP route.
- If you configure the LSR to use the platform label space, it maps labels for new routes and distributes them immediately to all peers without waiting for a label request message from an upstream peer or a label mapping message from a downstream peer. If you configure the LSR to use the interface label space, it sends labels only upstream when requested by an upstream peer; it does not send a label until it has received any necessary label from downstream.
- Use the **ip-profile** keyword to specify an IP profile to be used for creating IP interfaces over the LSPs.

- Example
 

```
host1(config)#mpls topology-driven-lsp
```
- Use the **no** version to halt topology-driven LSP creation.

### *mpls topology-driven-lsp ip-interfaces*

- Use to enable the creation of IP interfaces over LSPs if you are configuring base LSP connectivity, to enable the LSPs to be used by routing at the LSP ingress and for IP termination at the LSP egress. For stacked LSP connectivity, IP interfaces are created automatically on top of the stacked interface in the VRF where needed.
- You can specify LSP egress, LSP ingress, prefix lists, or access lists that describe the LSPs, or LSPs that go to host addresses. For example, you can issue the command with the **ingress** keyword and again with the **egress** keyword to specify different qualifications for creating IP interfaces on the LSP ingress versus the LSP egress.
- Example

```
host1(config)#mpls topology-driven-lsp ip-interfaces egress
host-only
```

- MPLS always creates an IP interface on the LSP egress for the router-id, even if you do not configure IP interfaces to be created for the LSPs.
- Example configuration when you require full LDP base-level connectivity:

Create an IP profile to be used by all LSPs:

```
host1(config)#interface loopback 1
host1(config-if)#ip address 10.10.10.1 255.255.255.240
host1(config-if)#no ip proxy-arp
host1(config-if)#ip router isis edged1
host1(config-if)#exit
host1(config)#profile ipedged1
host1(config-profile)#ip virtual-router edged1
host1(config-profile)#ip unnumbered loopback 1
```

Create an access list for full connectivity to all addresses:

```
host1(config)#access-list aledged1 permit any
```

Apply the profile and access list:

```
host1(config)#mpls topology-driven-lsp ip-profile ipedged1
host1(config)#mpls topology-driven-lsp ip-interfaces
access-list aledged1
```

Both ingress and egress routers on the LSPs will use the same access list. This configuration puts IP interfaces on all of the LSP ingresses and egresses; if you have 8000 routes (8000 LSPs), this configuration consumes 8000 IP interfaces.

A preferred alternative to this configuration is to create an access list that permits only the addresses for the LSPs for which you need connectivity, rather than all LSPs.

Another more commonly used—and recommended—configuration is to specify host-only interfaces. This method which does not require an access list and

puts IP interfaces on only the LSPs for the routes with host addresses. The following is an example of the commands for host-only support (note the IP profile is still required):

```
mpls topology-driven-lsp ip-profile ipedgel
mpls topology-driven-lsp ip-interfaces host-only (instead
of the access list)
```

- Use the **no** version to halt the creation of IP interfaces over all LSPs or the specified LSPs.

### ***mpls traffic-eng link-management timers periodic-flooding***

- Use to specify in seconds how often the current bandwidth value is flooded throughout the network.
- To turn periodic flooding off, set the value to 0.
- Example

```
host1(config)#mpls traffic-eng link-management timers
periodic-flooding 240
```

- Use the **no** version to restore the default value, 180.

### ***refresh-period***

- Use to specify the timeout period in milliseconds between generation of refresh messages.
- Specify a value in the range from 0–4294967295; the default value is 30,000 milliseconds.
- Together with the cleanup timeout factor, defines the RSVP tunnel timeout period. See *RSVP-TE Messages and Sessions* earlier in this chapter for more information.
- Example

```
host1(config-rsvp)#refresh-period 1000
```

- Use the **no** version to restore the default value, 30000.

## *Interface Configuration Tasks*

These tasks are performed at the major interface level on ATM, Ethernet (including VLANs), or POS interfaces. Creating or accessing an interface places the CLI in Interface Configuration mode. You can then configure MPLS options on that interface. The following sequence is arbitrary; you can perform these tasks in any order.



**Note:** Loop detection is always on in our MPLS implementation.

- 1 Enable MPLS on the interface.

```
host1(config-if)#mpls
```

- (Optional) Configure the interface label space via the VPI and VCI ranges.

```
host1(config-if)#mpls atm vpi range ldp 10 200
host1(config-if)#mpls atm vci range ldp 33 4000
```

Only ATM interfaces currently support the interface label space.

- Start LDP or RSVP-TE on the interface.

- Using the default values (an implicit *default* profile):

```
host1(config-if)#mpls ldp
or
```

```
host1(config-if)#mpls rsvp
```

- Using a previously created profile:

```
host1(config-if)#mpls ldp profile ldp5
or
```

```
host1(config-if)#mpls rsvp profile rsvp4
```

- (Optional) Configure total bandwidth available on the interface.

```
host1(config-if)#bandwidth 8192
```

- (Optional) Configure total bandwidth reservable for MPLS on the interface.

```
host1(config-if)#mpls bandwidth 4096
```

- (Optional) Specify thresholds that trigger bandwidth flooding when crossed by an increase or decrease in the total reservable bandwidth.

```
host1(config-if)#mpls traffic-eng flood thresholds up 15
host1(config-if)#mpls traffic-eng flood thresholds down 15
```

- (Optional) Specify the resource attributes for the interface so that tunnels can discriminate among interfaces.

```
host1(config-if)#mpls traffic-eng attribute-flags 0x000001f9
```

- (Optional) Configure an administrative weight for the interface that overrides the weight assigned by the IGP.

```
host1(config-if)#mpls traffic-eng administrative-weight 25
```

- (Optional) Specify an address to be advertised in LDP discovery hello messages as the transport address to be used by the peer router to establish the session.

```
host1(config-if)#mpls ldp discovery transport-address
10.10.25.42
```

When you have completed the interface configuration tasks, proceed to the next section, *Tunnel Configuration Tasks*.

### ***bandwidth***

- Use to specify the total bandwidth in kilobits per second available on the interface.
- Example

```
host1(config-if)#bandwidth 262144
```
- Use the **no** version to remove the admission control configuration (all internal CAC records) from the interface.

### ***mpls***

- Use to create, enable, disable, or delete MPLS on an interface.
- MPLS does not exist by default and must be created, either explicitly with this command, or implicitly with another **mpls** configuration command. If you create MPLS implicitly, for example by issuing the command **mpls bandwidth 32768**, MPLS remains disabled until you enable it, either by issuing the **mpls** command or the **no mpls disable** command.
- If you issue any other MPLS command, MPLS is disabled until you explicitly enable or create it using this command.
- Example

```
host1(config-if)#mpls
```
- Use the **mpls disable** version to stop MPLS on the interface while leaving the MPLS interface configuration intact.
- Use the **no mpls** version to halt MPLS on the interface and delete the MPLS interface configuration.
- Use the **no mpls disable** version to enable MPLS that was previously disabled or implicitly created on the interface.

### ***mpls atm vci range***

- Use to specify the range for virtual circuit identifiers on ATM major interfaces.
- An interface can support LDP, RSVP-TE, or both as the label distribution protocol. You can set VCI ranges independently for each protocol. If you do not specify a keyword, the range applies to both protocols.
- Example

```
host1(config-if)#mpls atm vci range ldp 45 4000
```
- Use the **no** version to remove the range. For both LDP and RSVP-TE, this results in an incomplete configuration.

### ***mpls atm vpi range***

- Use to specify the range for virtual path identifiers for LSPs on ATM major interfaces.
- An interface can support LDP, RSVP-TE, or both as the label distribution protocol. You can set VPI ranges independently for each protocol. If you do not specify a keyword, the range applies to both protocols.
- Example

```
host1(config-if)#mpls atm vpi range ldp 10 200
```
- Use the **no** version to remove the range. For RSVP-TE, this results in an incomplete configuration. For LDP, this results in an incomplete configuration only if no VCI range is configured.

### ***mpls bandwidth***

- Use to specify the total bandwidth in kilobits per second *reservable* for MPLS on the interface.
- Use the **mpls** version of the command for our implementation.
- Use the **ip rsvp** version of the command for compatibility with non-ERX implementations.
- Example

```
host1(config-if)#mpls bandwidth 32768
```
- Use the **no** version to restore the default value, 0.

### ***mpls disable***

- Use to disable a label distribution protocol—either LDP or RSVP-TE—on the interface.
- Example

```
host1(config-if)#mpls ldp disable
```
- Use the **no** version to reenable a label distribution protocol previously disabled on the interface.

### ***mpls ldp discovery transport-address***

- Use to specify the transport address advertised in LDP discovery hello messages. The peer router uses the transport address to establish the session TCP connection.
- Use the **interface** keyword to specify that IP address of the interface is advertised as the transport address in discovery hello messages sent via the interface.
- If you do not use this command to specify a transport address, the remote peer uses the source IP address from the hello messages it has received from the local peer for the local peer's transport address.

- Example

```
host1(config-if)#mpls ldp discovery transport-address
10.10.25.42
```
- Use the **no** version to halt advertisement of the transport address.

### ***mpls profile***

- Use to configure a label distribution protocol—either LDP or RSVP-TE—on the interface.
- You can specify a profile name. If you specify a profile not previously configured, it is created with the factory default settings. If you do not specify a profile, the factory-supplied default profile values are used.
- Example

```
host1(config-if)#mpls ldp profile ldp45
```
- The **interface** keyword is not supported when you issue this command from Interface Configuration mode.
- Use the **no mpls** version to delete the protocol on the interface.
- Use the **no mpls disable** version to reenab a protocol that was previously disabled on the interface.
- Use the **no mpls profile** version to revert to the default profile on the interface.

### ***mpls traffic-eng administrative-weight***

- Use to specify an administrative weight for the interface.
- For MPLS traffic-engineering purposes, this value supersedes any weight conferred upon the link by the IGP and can range from 0 to 4294967295.
- Example

```
host1(config-if)#mpls traffic-eng administrative-weight 150
```
- Use the **no** version to restore the default value, which matches the IGP-determined weight.

### ***mpls traffic-eng attribute-flags***

- Use to specify traffic-engineering attributes for an interface; attributes are compared with tunnel affinity bits to determine links eligibility for the tunnel.
- Specify a hexadecimal value in the range 0x0–0xFFFFFFFF. The attribute values have only the meaning that you assign to them; they serve to place the interface within one or more resource classes.
- Example

```
host1(config-if)#mpls traffic-eng attribute-flags 0x100F0010
```
- Use the **no** version to restore the default value, 0x0.

### ***mpls traffic-eng flood thresholds***

- Use to specify the thresholds that trigger the flooding of the current reservable bandwidth throughout the network.

- Bandwidth is flooded when the percentage of total reservable bandwidth increases past any up threshold or decreases past any down threshold.
- You can list more than one percentage; flooding is triggered by all percentages specified.
- Example
 

```
host1(config-if)#mpls traffic-eng flood thresholds up 25
host1(config-if)#mpls traffic-eng flood thresholds down 25
```
- Use the **no** version to restore the default values:
  - › For increases in bandwidth – 15, 30, 45, 60, 75, 80, 85, 90, 95, 97, 98, 99, 100
  - › For decreases in bandwidth – 100, 99, 98, 97, 96, 95, 90, 85, 80, 75, 60, 45, 30, 15

### *Tunnel Configuration Tasks*

Complete the following tasks to configure a tunnel interface. You must create the tunnel interface first and configure the tunnel endpoint last; you can perform all other tasks in any order.

- 1 Create the MPLS tunnel interface.

```
host1(config)#interface tunnel mpls:boston
```

- 2 (Optional) For topology-driven mode only, enable targeted tunnels to use the LDP LSPs created automatically.

```
host1(config-if)#tunnel mpls dynamic target
```

- 3 (Optional) Specify the unnumbered loopback interface for the tunnel. This step is *not* optional if you are using policies for data mapping or if you configure the LSP to announce its endpoint to IS-IS or OSPF.

```
host1(config-if)#ip unnumbered loopback 1
```

- 4 (Optional) Configure the LSP to announce its endpoint to an IGP (this step is sometime referred to as registering the endpoint).

```
host1(config-if)#tunnel mpls autoroute announce isis
```

- 5 (Optional) Specify a tunnel metric.

```
host1(config-if)#tunnel mpls autoroute metric absolute 100
```

- 6 (Optional) Configure the path options used for the tunnel.

```
host1(config-if)#tunnel mpls path-option 3 dynamic isis
```

- 7 Specify the label distribution protocol for this tunnel. (For topology-driven mode, the label distribution protocol is LDP and is not otherwise specified.)  

```
host1(config-if)#tunnel mpls label-dist cr-ldp
```
- 8 (Optional) Configure the bandwidth required for the tunnel.  

```
host1(config-if)#tunnel mpls bandwidth 1240
```
- 9 (Optional) Configure preemption hold or setup priority.  

```
host1(config-if)#tunnel mpls traffic-eng priority 4 4
```
- 10 (Optional) Configure resource class affinity.  

```
host1(config-if)#tunnel mpls traffic-eng affinity 0x1100  
mask 0xFFFF
```
- 11 (Optional) Configure retry timers options to apply to a specific tunnel to set up an LSP after a setup failure.  

```
host1(config-if)#tunnel mpls no-route retries 100  
host1(config-if)#tunnel mpls no-route retry-time 45  
host1(config-if)#tunnel mpls retries 250  
host1(config-if)#tunnel mpls retry-time 65
```
- 12 (Optional) Configure a value for the TTL field placed in the MPLS header when a label is first assigned to a packet.  

```
host1(config-if)#no tunnel mpls ip propagate-ttl local
```
- 13 (Optional) Stack the tunnel on top of a base tunnel.  

```
host1(config-if)#tunnel mpls base-tunnel eastcoast2
```
- 14 (Optional) Associate a text description with the tunnel.  

```
host1(config-if)#tunnel mpls description southshore
```
- 15 Configure the tunnel endpoint.  

```
host1(config-if)#tunnel destination 10.12.21.5
```

### ***interface tunnel***

- Use to create a tunnel interface for MPLS.
- You can specify that the tunnel be established in the routing space of a virtual router other than the current VR. If you specify another VR, all MPLS tunnel commands apply to the tunnel in that VR. If you do not specify another VR, tunnel commands apply to the current VR.
- Example, tunnel in current VR

```
host1(config)#interface tunnel mpls:5
```

- Example, tunnel in another VR

```
host1(config)#interface tunnel mpls:5
transport-virtual-router vr5
```

- You can specify tunnel stacking with this command. The following example configures tunnel *metro1* on top of tunnel *boston*:

```
host1(config)#interface tunnel mpls:boston.metro1
```

For VPNs sharing the same core tunnel, you use this format to specify each VPN associated with the tunnel, as in the following example:

```
host1(config)#interface tunnel mpls:5.1
host1(config)#interface tunnel mpls:5.2
```

- Use the **no** version to remove the tunnel interface.

### ***ip unnumbered***

- Use to specify the loopback interface used by the tunnel.
- Enables IP processing on the interface without a specific IP address.
- Creates the IP interface on the current virtual router.
- Example

```
host1(config-if)#ip unnumbered loopback 1
```

- Use the **no** version to remove IP processing on the interface.

### ***tunnel destination***

- Use to configure the tunnel endpoint for static tunnels.
- The IP address for the destination must be one of the following:
  - › The IP address of the interface that has MPLS enabled
  - › The router ID of the destination router
- Example

```
host1(config-if)#tunnel destination 10.12.21.5
```

- Use the **no** version to delete the endpoint.

### ***tunnel mpls affinity***

- Use to configure an affinity constraint on a given tunnel.
- The affinity value specifies the class of resources—resource attributes—associated with the tunnel. Affinity values range from 0x0 to 0xFFFFFFFF; the default is 0x0.
- Applying the mask to the affinity bits determines the value of the attribute flags that a link (interface) must have in order to be used by a tunnel. If a mask bit is one, the corresponding interface attribute flag must match the tunnel's corresponding affinity bit. If a mask bit is zero, the attribute flag does not have to match the tunnel's affinity bit. Mask values range from 0x0 to 0xFFFFFFFF; the default is 0x0000FFFF.
- In the current release, RSVP-TE does not include the affinity bits in its messages. As a result RSVP-TE cannot use any configured affinity restraint for

portions of tunnels beyond the ingress router's local area. CR-LDP does not have this restriction.

- Example 1

```
host1(config-if)#tunnel mpls affinity 0x000C3000 0xFFFFFFFF
```

matches only links configured with attribute flags 0x000C3000

- Example 2

```
host1(config-if)#tunnel mpls affinity 0x000C3000 0xFFFFFFFFE
```

matches only links configured with attribute flags 0x000C3000 or 0x000C3001

- Example 3

```
host1(config-if)#tunnel mpls affinity 0x000C3000 0xFFFFFFFFC
```

matches only links configured with attribute flags from 0x000C3000 to 0x000C3003

- Example 4

```
host1(config-if)#tunnel mpls affinity 0x000C3000 0xFFFFFFFF0
```

matches only links configured with attribute flags from 0x000C3000 to 0x000C300F

- Use the **no** version to delete the affinity constraint from the tunnel.

### ***tunnel mpls autoroute announce***

- Use to configure the LSP to register its endpoint (the egress router) with the configured routing protocol—BGP, IS-IS, or OSPF—so that the protocol can use the tunnel to determine routes.
- If you do not specify a routing protocol, the default behavior is to announce to both IS-IS and OSPF (but not BGP).
- If you have configured BGP/MPLS VPNs (*Chapter 3, Configuring BGP/MPLS VPNs*), you must use this command to announce tunnel endpoints to BGP.
- If you announce the endpoint to BGP and want BGP to express a preference for using the LSP, you must configure an absolute metric with the **tunnel mpls autoroute metric** command.

- Example

```
host1(config-if)#tunnel mpls autoroute announce isis
```

- Use the **no** version to disable endpoint announcements.

### ***tunnel mpls autoroute metric***

- Use to specify the tunnel metric. The value determines tunnel preference when there is more than one tunnel or native IP path to a tunnel endpoint. A lower value is preferred to a higher value.
- When you set up multiple tunnels, if the primary tunnel goes down, the existing tunnel with the lowest metric is used immediately.
- If you specify an absolute value from 1–65535, this value overrides the metric for the path provided by the IGP.

- If you specify a relative value from -10 to +10, this value is subtracted from (-) or added to (+) the metric for the path provided by the IGP.
- If you announce the endpoint to BGP and want BGP to express a preference for using the LSP, you must configure an absolute metric with this command.

- Example

```
host1(config-if)#tunnel mpls autoroute metric relative -5
```

- Use the **no** version to restore the default value, *relative 0*, meaning that the tunnel metric is the IGP value.

### ***tunnel mpls bandwidth***

- Use to configure a bandwidth constraint on a given tunnel in kilobits per second.
- Example

```
host1(config-if)#tunnel mpls bandwidth 2148
```

- Use the **no** version to delete the bandwidth constraint from the tunnel.

### ***tunnel mpls base-tunnel***

- Use to stack the tunnel on top of the specified base tunnel.
- Example

```
host1(config-if)#tunnel mpls base-tunnel eastcoast2
```

- Use the **no** version to remove the tunnel from on top of the base tunnel.

### ***tunnel mpls description***

- Use to associate a text description with the tunnel.
- Example

```
host1(config-if)#tunnel mpls description boston2dc
```

- Use the **no** version to delete the description.

### ***tunnel mpls dynamic target***

- Use to designate a targeted tunnel that MPLS will dynamically stack above an LDP base LSP.
- Example

```
host1(config-if)#tunnel mpls dynamic target
```

- Use the **no** version to stop the tunnel from being dynamically stacked over an LDP base LSP.

### ***tunnel mpls ip propagate-ttl***

- Use to configure the time-to-live field placed in the MPLS header when a label is first added to an IP packet.
- This command overrides settings configured at the global level with the **mpls ip propagate-ttl** command.

- This command is enabled by default, causing the TTL field to be copied from the IP packet header and enabling the **traceroute** command to show all the hops in the network. This command also propagates the label TTL into the IP header at the tunnel egress.
- This command applies only to the tunnel ingress. Do not issue this command on the router hosting the tunnel egress.
- Example

```
host1(config-if)#tunnel mpls ip propagate-ttl
```
- Use the **default** version to revert to the settings configured at the global configuration level. If the command was not issued at the global level, then the interface setting reverts to the global default, causing the TTL field to be copied from the IP packet header and enabling the **traceroute** command to show all the hops in the network. The **default** version propagates the label TTL into the IP header at the tunnel egress.
- Use the **no** version to specify a fixed TTL value of 255 for the first label of the MPLS header and cause the structure of the MPLS network (all packets) to be hidden from a **traceroute** command.

You can specify the types of packets to be hidden from **traceroute** by using the following keywords:

- › **forwarded** – **traceroute** cannot show the hops for forwarded packets; this most common application of the command enables you to hide the structure of the MPLS network from your customers:

```
host1(config-if)#no tunnel mpls ip propagate-ttl forwarded
```

- › **local** – **traceroute** cannot show the hops for local packets

```
host1(config-if)#no tunnel mpls ip propagate-ttl local
```

### ***tunnel mpls label-dist***

- Use to specify the label distribution protocol as CR-LDP or RSVP-TE.
- For topology-driven mode, the tunnels are implicitly set as LDP and you do not configure the label distribution protocol with this command.

- Example

```
host1(config-if)#tunnel mpls label-dist rsvp-te
```

- Use the **no** version to remove the label distribution protocol.

### ***tunnel mpls lsp retries***

- Use to specify the number of attempts that will be made to set up an LSP for CR-LDP and RSVP-TE after a failure *other* than one due to no available route.
- The default value of 0 means the attempts will be made until successful.

- Example

```
host1(config-if)#tunnel mpls retries 1275
```

- Use the **no** version to restore the default value, 0.

### ***tunnel mpls lsp retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for CR-LDP and RSVP-TE after a failure *other* than one due to no available route.
- Example

```
host1(config-if)#tunnel mpls retry-time 15
```
- Use the **no** version to restore the default value, 5.

### ***tunnel mpls no-route retries***

- Use to specify the number of attempts that will be made to set up an LSP for CR-LDP and RSVP-TE after a failure due to no available route.
- The default value of 0 means the attempts will be made until successful.
- Example

```
host1(config-if)#tunnel mpls no-route retries 3200
```
- Use the **no** version to restore the default value, 0.

### ***tunnel mpls no-route retry-time***

- Use to specify the interval in seconds between attempts to set up an LSP for CR-LDP and RSVP-TE after a failure due to no available route.
- Example

```
host1(config-if)#tunnel mpls no-route retry-time 45
```
- Use the **no** version to restore the default value, 5.

### ***tunnel mpls path-option***

- Use to specify the path options for the tunnel. You can configure one or more path options—each identified by a unique number—for a given tunnel.
- Options include whether the path is dynamic or explicit and whether hop-by-hop, IS-IS, or OSPF routing is used.
- Options with a lower number are preferred; path option 1 is the most preferred. If an LSP goes down on the currently used path option, the path option with the next highest preference is used.
- Example

```
host1(config-if)#tunnel mpls path-option 3 dynamic isis
```
- Use the **no** version to delete the path options.

### ***tunnel mpls priority***

- Use to configure a priority for a given tunnel, ranging from the highest priority of 0 to the lowest priority of 7.
- You can configure a setup priority or both a setup priority and a hold priority for the tunnel:
  - › Setup priority – priority of an LSP while it is being established; default value is 4

- › Hold priority – priority of an LSP once it has been established; default value is equal to the specified setup priority

If insufficient resources exist for a new LSP to be established, its setup priority is evaluated against the hold priorities of existing LSPs. If the new LSP has a higher priority, it preempts the resources from the lower-priority existing LSP(s) and is established.

- A setup priority of 0 can preempt all nonzero hold priorities.
- The setup priority cannot be better (lower numerically) than the hold priority. For example, if the setup priority for a tunnel is 2, the hold priority must be 2, 1, or 0.
- Example

```
host1(config-if)#tunnel mpls priority 3 2
```
- Use the **no** version to delete the priority from the tunnel.

### *Tunnel Profile Configuration Tasks*

If you anticipate having tunnels that will share the same configuration, you can reduce your configuration time by using tunnel *profiles*.

To configure a tunnel profile:

- 1 Enter Tunnel Profile Configuration mode.

```
host1(config)#mpls tunnels profile Lisbon
```

- 2 Use the commands described in steps 3 through 6 in the previous section, *Tunnel Configuration Tasks*, to configure the following profile settings:

- Endpoint announcement
- Tunnel metric
- Path options
- Label distribution protocol

- 3 Configure the tunnel endpoint.

- For static tunnels

```
host1(config-tunnelprofile)#tunnel destination 10.1.2.5  
10.1.2.6
```

All tunnels to the specified destination(s) are configured with the profile settings.

- For dynamic tunnels

```
host1(config-tunnelprofile)#tunnel destination isis-level-2
access-list madrid3
```

When an endpoint is dynamically learned from the specified routing protocol, MPLS searches its tunnel profiles for a match. The dynamic tunnel is established using the settings from the first matching profile

- 4 Configure an associated IP profile.

```
host1(config-tunnelprofile)#tunnel ip profile IPLisbon
```

See *ERX Routing Protocols Configuration Guide, Vol. 1, Chapter 2, Configuring IP*, for information on creating IP profiles.

- 5 Configure the bandwidth required for the tunnel.

```
host1(config-tunnelprofile)#tunnel mpls bandwidth 1240
```

- 6 Configure preemption hold or setup priority.

```
host1(config-tunnelprofile)#tunnel mpls priority 4 4
```

- 7 Configure resource class affinity.

```
host1(config-tunnelprofile)#tunnel mpls affinity 0x1100 mask
0xFFFF
```

- 8 Stack the tunnel on top of a base tunnel.

```
host1(config-if)#tunnel mpls base-tunnel eastcoast2
```

- 9 Associate a text description with the tunnel.

```
host1(config-if)#tunnel mpls description southshore
```

### ***mpls tunnels profile***

- Use to create a tunnel profile for MPLS.
- A tunnel profile is used for all tunnels sharing the same configuration.
- Example

```
host1(config)#mpls tunnels profile Paris
```

- Use the **no mpls tunnels profile** version to delete the tunnel profile. Use the **no mpls tunnels profile disable** version to reenables tunnels previously disabled.

### ***tunnel destination***

- Use to configure the profile to accept tunnel endpoints (destinations) associated with the profile.
- If you specify that the endpoints are to be learned from IS-IS or OSPF, tunnels are created when the destinations are learned from the specified IGP. you can filter learned addresses by specifying an access list or a prefix list.
- If you specify a sequence of one or more individual IP addresses as endpoints, tunnels are created as soon as the destination addresses are configured.
- If you specify one or more IP addresses for the destination(s), each address must be one of the following:
  - › The IP address of the interface that has MPLS enabled
  - › The router ID of the destination router
- Examples

```
host1(config-tunnelprofile)#tunnel destination isis-level-2
prefix-list tunnel5
host1(config-tunnelprofile)#tunnel destination 10.12.25.1
10.12.25.2
```
- Use the **no** version to delete the endpoints.

### ***tunnel ip profile***

- Use to assign an IP profile to the MPLS tunnels in the tunnel profile.
- Example

```
host1(config-tunnelprofile)#tunnel ip profile euro5
```
- Use the **no** version to remove the IP profile from the tunnel profile.

### ***tunnel mpls base-tunnel***

- Use to stack the tunnel on top of the specified base tunnel.
- Example

```
host1(config-tunnelprofile)#tunnel mpls base-tunnel
eastcoast2
```
- Use the **no** version to remove the tunnel from on top of the base tunnel.

### ***tunnel mpls description***

- Use to associate a text description with the tunnel.
- Example

```
host1(config-tunnelprofile)#tunnel mpls description
boston2dc
```
- Use the **no** version to delete the description.

## Explicit Routing

---

MPLS offers two options for selecting routing paths:

- Hop-by-hop routing
- Explicit routing

In explicit routing, the route the LSP takes is defined by the ingress node. The path consists of a series of hops defined by the ingress LSR. Each hop can be a traditional interface, an autonomous system, or an LSP. A hop can be *strict* or *loose*.

A *strict* hop must be directly connected (that is, adjacent) to the previous node in the path. A *loose* hop is not necessarily directly connected to the previous node; whether it is directly connected is unknown.

The sequence of hops comprising an explicit routing LSP may be chosen in either of the following ways:

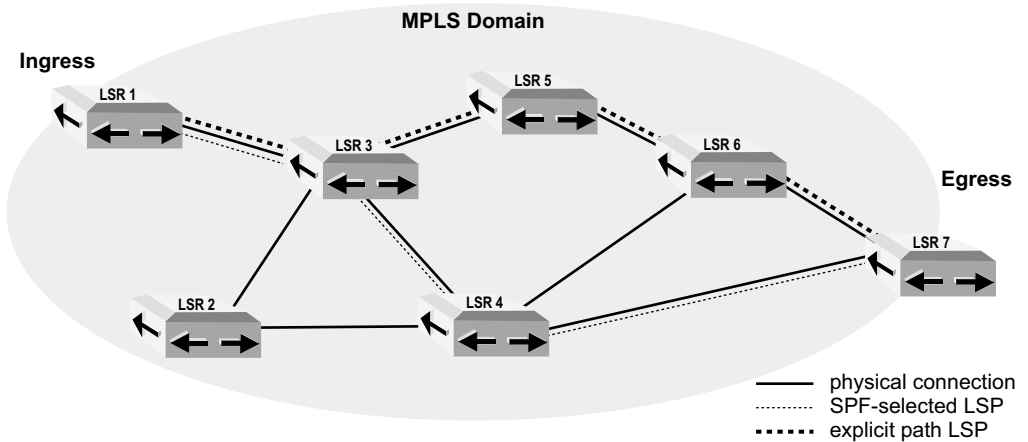
- Through a user-defined configuration, resulting in *configured* explicit paths. When you create the explicit route, you must manually configure each hop in the routing path (LSP).
- Through a routing protocol–defined configuration, resulting in *dynamic* explicit paths. When the routing protocol (IS-IS or OSPF) creates the explicit path, it makes use of the topological information learned from a link state database in order to compute the entire path, beginning at the ingress node and ending at the egress node.

Consider the MPLS domain shown in Figure 2-7. Without explicit path routing, the tunnel is created hop by hop along the following path:

LSR 1 -> LSR 3 -> LSR 4 -> LSR 7

Suppose LSR 5 and LSR 6 are underused and LSR 4 is overused. In this case you might choose to configure the following explicit path because it forwards the data better than the hop-by-hop path:

LSR 1 -> LSR 3 -> LSR 5 -> LSR 6 -> LSR 7



**Figure 2-7** Explicit routing in an MPLS domain

### Defining Configured Explicit Paths

You can create explicit routing paths *manually* by configuring an explicit path with a name and a series of addresses (hops) from ingress to egress.

To manually configure explicit routing:

- 1 Enter Explicit Path Configuration mode (config-expl-path).
 

```
host1(config)#mpls explicit-path name xyz
host1(config-expl-path)#
```
- 2 Configure the hops in the LSP.
- 3 Set the next hop (if need be) at a particular index in the explicit path.
- 4 Add the next hop (if need be) after a particular index in the explicit path.
- 5 Enable the explicit path.



**Note:** To prevent a partially configured explicit path from being used, do not enable it until you have finished configuring or modifying the path.

- 6 (Optional) List the currently configured explicit path.

### **append-after**

- Use to add a next hop after a particular index in the explicit path.
- Sequence (index) numbers after this index adjust automatically.

- Example

```
host1(config-expl-path)#append-after 5 next-address
192.168.47.22
```
- There is no **no** version.

### *index*

- Use to set a next hop at a particular index in the explicit path.
- Example

```
host1(config-expl-path)#index 5 next-address 172.18.100.5
```
- Use the **no** version to remove the next hop from the index.

### *list*

- Use to list the currently configured explicit path (optionally starting at a particular index defined with the **index** command).
- Example

```
host1(config-expl-path)#list 5
```
- There is no **no** version.

### *mpls explicit-path*

- Use to define an explicit path by name and also to enable or disable an explicit path.
- Also accepts the **ip** keyword instead of **mpls** for compatibility with non-ERX implementations.
- Examples

```
host1#mpls explicit-path name xyz
host1#mpls explicit-path name xyz enable
```
- Use the **no** version to delete the specified explicit path.

### *next-address*

- Use to configure an IPv4 hop at the end of the explicit path.
- You can specify a loose node, which indicates that the node is not necessarily directly connected (adjacent) to the previous node in the path. If you do not specify loose, the configuration defaults to strict, indicating that the node is directly connected to the previous node.
- Example

```
host1(config-expl-path)#next-address 10.10.9.2
```
- There is no **no** version.

## Specifying Configured Explicit Paths on a Tunnel

After you have defined a configured explicit path, you can configure the path on a tunnel.

To configure explicit routing on a tunnel:

- 1 Create an MPLS tunnel.

```
host1(config)#interface tunnel mpls:1
```

- 2 Set the path option.

```
host1(config-if)#tunnel mpls path-option 1 explicit name xyz
```

### **interface tunnel**

- Use to create a tunnel interface for MPLS.
- You can specify that the tunnel be established in the routing space of a virtual router other than the current VR. If you specify another VR, all MPLS tunnel commands apply to the tunnel in that VR. If you do not specify another VR, tunnel commands apply to the current VR.

- Example, tunnel in current VR:

```
host1(config)#interface tunnel mpls:5
```

- Example, tunnel in another VR:

```
host1(config)#interface tunnel mpls:5  
transport-virtual-router vr5
```

- You can specify tunnel stacking with this command. The following example configures tunnel *metro1* on top of tunnel *boston*:

```
host1(config)#interface tunnel mpls:boston.metro1
```

For VPNs sharing the same core tunnel, you use this format to specify each VPN associated with the tunnel, as in the following example:

```
host1(config)#interface tunnel mpls:5.1  
host1(config)#interface tunnel mpls:5.2
```

- Use the **no** version to remove the tunnel interface.

### **tunnel mpls path-option**

- Use to specify the path options for the tunnel. You can configure one or more sets of path options—each identified by a unique number—for a given tunnel.
- Options with a lower number are preferred; path option 1 is the most preferred. If an LSP goes down on the currently used path option, the path option with the next highest preference is used.

- Examples – configured and dynamic tunnels:

```
host1(config-if)#tunnel mpls path-option 1 explicit name xyz  
host1(config-if)#tunnel mpls path-option 2 dynamic isis
```

- Use the **no** version to delete the path options.

### *Configuring Dynamic Explicit Paths on a Tunnel*

You can create explicit routing paths *dynamically* via a routing protocol. IS-IS and OSPF both currently support explicit routing.

To configure dynamic explicit routing:

- 1 Create an MPLS tunnel.

```
host1(config)#interface tunnel mpls:bilbao5
```

- 2 Set the path option.

```
host1(config-if)#tunnel mpls path-option 2 dynamic isis
```

To configure MPLS dynamic explicit path routing, refer to the commands and guidelines in the previous section, *Specifying Configured Explicit Paths on a Tunnel*.

### *Monitoring Explicit Paths*

For information on monitoring MPLS explicit routing, refer to the following sections in this chapter:

- *Monitoring MPLS*
- *Configuring IGP's and MPLS*

## Configuring Virtual Private Networks

---

MPLS supports the following models of virtual private networks (VPNs):

- Virtual router model – VPN-specific MPLS tunnels are established and serve as virtual interfaces in the VPNs. Routing protocols distribute the VPN routes over the virtual interfaces. The route distribution task belongs to the VPNs and is transparent to ISPs. Only static routes are supported in this release.
- BGP/MPLS (RFC 2547) model – VPN routes and their associated MPLS labels are distributed via enhanced BGP. This model employs label stacking. BGP must request a label from MPLS that it can then associate with a VPN. The tunnel endpoint provides a label to BGP, which BGP then announces in its update messages.

Only the first model is discussed in this section. For detailed information on BGP/MPLS VPNS, see *Chapter 3, Configuring BGP/MPLS VPNS*.

### *Virtual Router Model*

Without MPLS, you face the addressing problem of announcing every single prefix in the VPN. Mapping an MPLS label to a routing table

enables you to have a single label represent every address in the VPN's routing table.

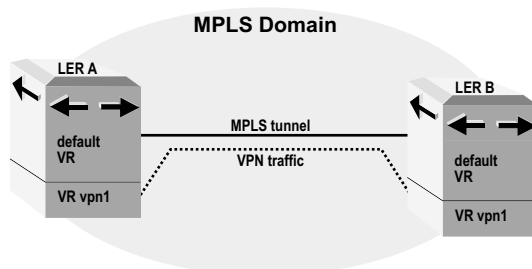
VPNs have their own terminology. A VPN runs between provider edge routers (PEs) in the service provider core. These PEs correspond to ingress and egress LERs. Between the PEs, the tunnel runs through zero or more provider core routers (Ps), which correspond to LSRs.

The MPLS tunnel runs from a virtual router (VR) on one PE (LER) to a VR on another PE (LER). The tunnel is VPN specific; you configure one tunnel per VPN per pair of PEs. The tunnel is used as a virtual interface within the VPN. You configure your IGP (IS-IS or OSPF) to run in a VR, typically the default VR, on the LERs. By virtue of running the IGP, this VR is aware of the complete system topology. For that reason you configure your base transport tunnels in this virtual router.

You must configure a separate VR on each system to host the VPN. The VPN VRs see only the private network, not the complete system topology, and have no connectivity until a tunnel goes up. They are therefore not appropriate for hosting the MPLS tunnel. The VPN traffic runs between these VRs on the MPLS tunnel as shown in Figure 2-8.



**Note:** You can configure the tunnel on the default VR, or any other VR, but it must be the same VR on both ends. The examples in this section use the default VR.

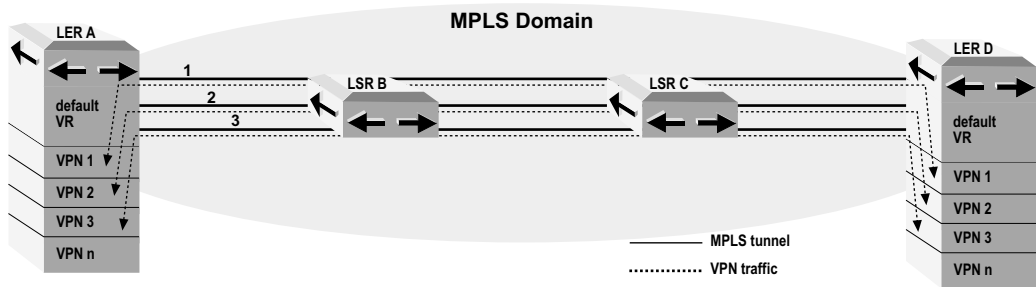


**Figure 2-8** MPLS virtual private network

You must create a separate MPLS tunnel for each VPN you configure. For example, you might configure a VPN on VR 2 to send traffic through the default VR that hosts the MPLS tunnel. The default VR then signals a tunnel for the VPN to the other side, passing a VPN identifier so that the other end knows to terminate the traffic in VR 2.

A single label is applied to all traffic along the MPLS tunnel. This label is swapped out at each LSR to forward the traffic to the tunnel endpoint. If you configured multiple VPNs to use the same core tunnel, there would be no way to determine which traffic belonged to which VPNs; you must

establish an additional tunnel for each additional VPN. Figure 2-9 shows a system with three VPNs running between LER A and LER D: VPN 1 on tunnel 1, VPN 2 on tunnel 2, and VPN 3 on tunnel 3. Labels for traffic on each tunnel are swapped at each node on the LSP.



**Figure 2-9** MPLS virtual private networks without label stacking

Compared with the BGP/MPLS model, the virtual router model enables you to control the routing inside a VPN. The drawback to the virtual router model is that it requires more configuration than the BGP/MPLS model. Opinion is divided about the comparative scalability (other than the configuration aspect of scalability) of the two models. The virtual router model can effectively meet your needs if you have only a few VPNs, or if some of the node devices cannot handle label stacking, or if you do not want the overhead of implementing BGP.

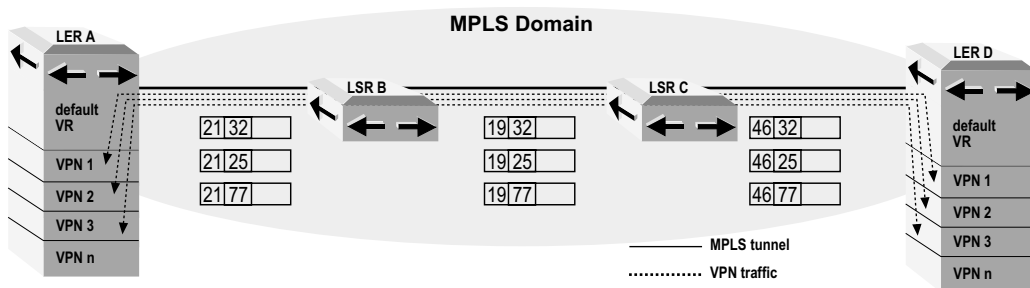
### VPN Stacking

Label stacking provides the means to distinguish between traffic flows from multiple VPNs across the same tunnel (between the same PEs). BGP MPLS and targeted hello messages can both provide a second-level label that identifies an individual VPN. As a result, a single core tunnel, typically running in the default VRs on two edge nodes, can service as many VPNs as desired while still handling basic, single-level label traffic as before. Stacking VPNs in this manner reduces the number of VPN tunnels within the public routing space and improves scalability. Refer to *Label-Switching Routers* earlier in this chapter for detailed information on label stacking. The current release supports only CR-LDP for stacking.

Each VPN that you configure gets a label from the platform label space to identify the VPN. When data is forwarded from the VPN through the MPLS tunnel, the default VR prepends a different, first-level label for the data flow from either the platform or interface label space. As the packet heads through the core tunnel, this first-level label is the one that is examined and swapped to move the packet to its destination.

The second-level label is unexamined and unchanged until the packet reaches the tunnel endpoint. At the endpoint, MPLS looks for a label stacked below the first level. MPLS continues processing the packet based on the second-level label. The second-level label identifies the VPN to which the packet belongs, so MPLS does a lookup in the VPN's routing table to determine what to do with the packet.

Consider the system shown in Figure 2-10. VPNs 1, 2, and 3 all share the same MPLS core tunnel. The second-level label for VPN 1 is 32; for VPN 2 it is 25, and for VPN 3 it is 77. These labels uniquely identify each VPN. Prependded to these labels in each data packet is a first-level label that is switched at each link in the LSP to forward the packets between the edge nodes, LER A and LER D. At LER D, the first-level label is popped, and the second-level label is used to identify which VPN owns the packet and to determine what further action to take regarding the packet.



**Figure 2-10** MPLS virtual private networks with label stacking

### BGP

BGP must request a label from MPLS that it can then associate with a VPN. The tunnel endpoint provides a label to BGP, which BGP then announces in its update messages.

For example, if the endpoint provides label 32 for VPN 1, the ingress node learns that 32 is paired with VPN 1 when BGP floods its network with that information. The ingress node then creates a label stack with label 32 at the second level for traffic on that VPN. This functionality is provided in extensions to BGP and requires no configuration in MPLS. For more information on BGP, see *Chapter 1, Configuring BGP Routing*. For more information on BGP/MPLS VPNs, see *Chapter 3, Configuring BGP/MPLS VPNs*.

### CR-LDP

If you do not want the overhead of running BGP, and especially if you are not already running BGP, you might prefer to use the CR-LDP method.

You might want to establish individual base core tunnels; this facilitates serving traffic with different QoS levels.

For example, you could set up three core tunnels corresponding to Gold, Silver, and Bronze levels of service. All of your VPNs could share these core tunnels depending on the QoS guaranteed for traffic within the VPNs. All Gold-level traffic would use the Gold core tunnel, regardless of which VPN carries the traffic; all Silver-level traffic would use the Silver core tunnel; and all Bronze-level traffic would use the Bronze core tunnel. This scheme enables you to minimize the number of tunnels you must establish through your network, in this case three, in order to serve all of your VPNs.

In addition to directly connected, adjacent peer sessions, CR-LDP can establish targeted sessions to create remote adjacencies with nodes that are not directly connected. From the default VR at the ingress, MPLS targets the egress address, sending the VPN information in the targeted message. The egress node maps a label to the VPN—for example, label 32 for VPN 1—and returns this to the ingress node, where it is employed as the second-level label.

Label stacking enables the creation of large-scale networks with fewer tunnels than those required without label stacking. You can establish one tunnel per class of service rather than one tunnel per VPN.

### *Configuring VPNs Using the VR Model*

To configure an MPLS VPN between two physical routers using the virtual router model:

- 1 Configure the VPN ID associated with the virtual router for the VPN on both systems.

On router 1:

```
host1(config)#virtual-router vpn3  
host1:vpn3(config)#ip vpn-id oui 20 index 10.4.4.4
```

On router 2:

```
host2(config)#virtual-router vpn3  
host2:vpn3(config)#ip vpn-id oui 20 index 10.4.4.4
```

- 2 Configure the MPLS tunnel used for the VPN in the core routing space (usually the default VR), and specify that it is used for the VPN by configuring a VPN ID for the tunnels.
- You can configure the tunnel from the context of the default virtual routers as follows:

On router 1:

```
host1(config)#virtual-router default
host1(config)#interface tunnel mpls:1
host1(config-if)#tunnel label-dist cr-ldp
host1(config-if)#tunnel mpls vpn-id oui 20 index 10.4.4.4
host1(config-if)#tunnel destination 172.20.1.1
```

On router 2:

```
host2(config)#virtual-router default
host2(config)#interface tunnel mpls:1
host2(config-if)#tunnel label-dist cr-ldp
host2(config-if)#tunnel mpls vpn-id oui 20 index 10.4.4.4
host2(config-if)#tunnel destination 172.20.1.2
```

- Alternatively, you can configure the tunnel from the context of the *VPN* virtual routers, remembering that the tunnel itself runs between the *default* VRs. To do so, you must use the **transport-virtual-router** option when you configure the tunnel:

On router 1:

```
host1(config)#virtual-router vpn3
host1:vpn3(config)#interface tunnel mpls:1
transport-virtual-router default
host1:vpn3(config-if)#tunnel label-dist cr-ldp
host1:vpn3(config-if)#tunnel mpls vpn-id oui 20 index
10.4.4.4
host1:vpn3(config-if)#tunnel destination 172.20.1.1
```

On router 2:

```
host2(config)#virtual-router vpn3
host2:vpn3(config)#interface tunnel mpls:1
transport-virtual-router default
host2:vpn3(config-if)#tunnel label-dist cr-ldp
host2:vpn3(config-if)#tunnel mpls vpn-id oui 20 index
10.4.4.4
host2:vpn3(config-if)#tunnel destination 172.20.1.2
```

- 3 In the context of the virtual router for the VPN, configure an IP interface on top of the MPLS tunnel that is in the core routing space.

On router 1:

```
host1(config)#virtual-router vpn3
host1:vpn3(config)#interface tunnel mpls:1
host1:vpn3(config-if)#ip unnumbered loopback 0
```

On router 2:

```
host2(config)#virtual-router vpn3
```

```
host2:vpn3(config)#interface tunnel mpls:1
host2:vpn3(config-if)#ip unnumbered loopback 0
```

- 4 In the context of the virtual router for the VPN, configure static routes using the IP interface as the next hop interface.

On router 1:

```
host1(config)#virtual-router vpn3
host1:vpn3(config)#ip route 172.20.3.0 255.255.255.0
    tunnel mpls:1
```

On router 2:

```
host2(config)#virtual-router vpn3
host2:vpn3(config)#ip route 172.20.4.0 255.255.255.0
    tunnel mpls:1
```

For each additional VPN that you want to create, you must repeat steps 1–4, creating a new core tunnel for each new VPN.

### Configuring VPNS with CR-LDP

To configure multiple VPNs to share an LSP with CR-LDP:

Perform steps 1–4 as for a basic MPLS VPN in *Configuring VPNs Using the VR Model*, except that in step 2 you must make the following changes:

- Do not associate the tunnel with the VPN; that is, omit the **tunnel mpls vpn-id** commands on both router 1 and router 2.
- Issue the following additional commands, from the context of the default virtual router or the VPN virtual router as appropriate:
  - a Create send and receive lists for targeted remote peers, and configure the lists on each router.

On router 1:

```
host1:vpn3(config)#access-list boston5 permit host
    172.20.3.0
host1:vpn3(config)#mpls ldp targeted-hello send list boston5
host1:vpn3(config)#mpls ldp targeted-hello receive list
    boston5
```

On router 2:

```
host2:vpn3(config)#access-list concord3 permit host
    172.20.4.0
host2:vpn3(config)#mpls ldp targeted-hello send list
    concord3
host2:vpn3(config)#mpls ldp targeted-hello receive list
    concord3
```

- b Specify the VPN associated with the tunnel when you create the tunnel interface, and associate the tunnel with a VPN ID.

On router 1:

```
host1:vpn3(config)#interface tunnel mpls:1.3
host1:vpn3(config-if)#tunnel mpls vpn-id oui 20 index
10.4.4.4
```

On router 2:

```
host2:vpn3(config)#interface tunnel mpls:1.3
host2:vpn3(config-if)#tunnel mpls vpn-id oui 20 index
10.4.4.4
```

This example creates tunnel 1 and associates it with VPN 3.

- For each additional VPN that you want to share the MPLS tunnel, you must do the following:
  - > Repeat step 1 with a VPN ID for the new VPN.
  - > Associate the new VPN ID with the tunnel, as in step 2.
  - > Repeat steps 3 and 4 for the new VPN.

### ***interface tunnel***

- Use to create a tunnel interface for MPLS.
- You can specify that the tunnel be established in the routing space of a virtual router other than the current VR. If you specify another VR, all MPLS tunnel commands apply to the tunnel in that VR. If you do not specify another VR, tunnel commands apply to the current VR.
- Example, tunnel in current VR:

```
host1(config)#interface tunnel mpls:5
```
- Example, tunnel in another VR:

```
host1(config)#interface tunnel mpls:5
transport-virtual-router vr5
```
- You can specify tunnel stacking with this command. The following example configures tunnel *metro1* on top of tunnel *boston*:

```
host1(config)#interface tunnel mpls:boston.metro1
```

For VPNs sharing the same core tunnel, you use this format to specify each VPN associated with the tunnel, as in the following example:

```
host1(config)#interface tunnel mpls:5.1
host1(config)#interface tunnel mpls:5.2
```

and so on.

- Use the **no** version to remove the tunnel interface.

***ip vpn-id***

- Use to associate a VPN ID with the virtual router.
- Configure the VPN ID identically on the VRs at each end of the VPN.
- Example

```
host1(config)#ip vpn-id oui 5 index 10.32.5.1
```

- Use the **no** version to remove the VPN ID from the virtual router.

***tunnel mpls vpn-id***

- Use to associate a tunnel with a VPN.
- Specify the same values for the VPN ID that you specified when you associated a VR with the VPN ID.
- Example

```
host1(config-if)#tunnel mpls vpn-id oui 5 index 10.32.5.1
```

- Use the **no** version to remove the VPN ID from the virtual router.

***Targeted Sessions to Directly Connected Peers***

Targeted sessions are typically not configured for directly connected peers. Configuring a targeted session to a directly connected peer on an interface that uses the platform label space requires an additional configuration step to work properly. For example, you might set up a one-hop targeted CR-LDP session and enable CR-LDP on a major interface that is also adjacent to the same peer.

If an existing link level session uses the same values as the targeted session for the LSR's router ID and the label space ID, then CR-LDP cannot distinguish between the link session and the targeted session. The two sessions will interfere with each other and cause unpredictable CR-LDP behavior.

You can avoid this situation by using the **mpls ldp discovery transport-address** command to set the transport address of the link level sessions to something other than the LSR's router ID, such as the local interface address. This enables CR-LDP to differentiate between directly connected link sessions and targeted sessions.

## Configuring IGPs and MPLS

---

You can use the **tunnel mpls autoroute announce** command to configure a tunnel to announce its endpoint to IS-IS or OSPF so that the IGP can then use the LSP as a shortcut to a destination based on the LSP's metric.



**Note:** This section discusses IS-IS and OSPF; for information on BGP and MPLS, see *Configuring Virtual Private Networks in this chapter and Chapter 3, Configuring BGP/MPLS VPNs*.

If no tunnels are present, the IGP calculates the shortest path to a destination by using the shortest path first (SPF) algorithm. The results are represented by the destination node, next-hop address, and output interface, where the output interface is a physical interface. With traffic engineering, the IGP can include MPLS label-switched paths (LSPs or tunnels) as part of the SPF calculation.

If you configure an LSP to be announced to the IGP with a certain metric, the LSP appears as a logical interface directly connected to the LSP endpoint. The IGP can consider the LSP as a potential output interface for the LSP endpoint and for destinations beyond the endpoint. In this case, the SPF computation results are represented by the destination node and the output LSP, effectively using the LSP as a shortcut through the network to the destination.

By default, IS-IS and OSPF always use the MPLS tunnel to reach the tunnel endpoint. Best paths determined by SPF calculations are not considered. You can enable the consideration of best paths by issuing the **mpls spf-use-any-best-path** command. This command causes the IGP to evaluate the LSP as it would any other path. The IGP then either forwards traffic along the best path (which might be the MPLS tunnel), or load-balances between the MPLS tunnel and another path.

The default behavior applies only to reaching the tunnel endpoint itself. For prefixes downstream of the tunnel endpoint, the value of the tunnel metric always determines whether the IGP uses the LSP or the native path, or load-balances between the native path and one or more LSPs.

The tunnel metric can be absolute or relative. An *absolute* metric indicates there is no relationship to the underlying IGP cost. A *relative* metric is added to or subtracted from the underlying IGP shortest path cost.

**Example 1** The following commands announce the tunnel to OSPF and specify a relative metric of -2:

```
host1(config-if)#tunnel mpls autoroute announce ospf
host1(config-if)#tunnel mpls autoroute metric relative -2
```

By default, the LSP is preferred to reach the tunnel endpoint. OSPF will treat this LSP as having a metric of 2 less than the shortest path metric it has calculated. The LSP is therefore also preferred over other paths to prefixes beyond the tunnel endpoint.

**Example 2** The following commands announce the tunnel to OSPF, specify an absolute metric of 25, and configure OSPF to enable the consideration of SPF best paths:

```
host1(config-if)#tunnel mpls autoroute announce ospf
host1(config-if)#tunnel mpls autoroute metric absolute 25
...
host1(config-router)#mpls spf-use-any-best-path
```

OSPF will consider this metric in its SPF calculations for traffic to the tunnel endpoint as well as beyond the endpoint. Traffic will be routed via this LSP only if the other calculated paths have higher metrics.

### *Configuring the IGP for Traffic Engineering*

For both IGPs, you must issue two commands to enable the IGP to support traffic engineering. Refer to the *ERX Routing Protocols Configuration Guide, Vol. 1, Chapter 8, Configuring IS-IS* and *ERX Routing Protocols Configuration Guide, Vol. 1, Chapter 7, Configuring OSPF* for more information on using these commands.

- IS-IS – Enable the flooding of MPLS traffic-engineering link information into the specified IS-IS level with the **mpls traffic-eng** command. You must also specify a stable router interface with the **mpls traffic-eng router-id** command.

MPLS traffic engineering also requires that IS-IS generate the new-style TLVs that enable wider metrics. Use the **metric-style wide** command to generate the new-style TLVs. If you are using some IS-IS routers that still do not understand the new-style TLVs, use the **metric-style transition** command.

- OSPF – Enable OSPF areas for traffic engineering with the **mpls traffic-eng area** command. OSPF generates opaque LSAs—also known as type-10 opaque link area link states—to flood the traffic-engineering information to the specified area. OSPF builds a traffic-engineering database that it uses in the calculation of shortest path to destinations that satisfy specified traffic-engineering constraints. As with IS-IS, you must also specify a stable router interface with the **mpls traffic-eng router-id** command.

When you configure a node as the downstream endpoint of an LSP, you must provide a stable interface as the router ID for the endpoint. Typically you select a loopback interface because of its inherent stability. Use the **mpls traffic-eng router-id** command to designate the router as TE capable and to specify the router ID. For all tunnels that end at this node, set the tunnel destination to the destination node's traffic-engineering

router identifier, because the traffic-engineering topology database at the tunnel ingress uses that for its path calculation.

### *Monitoring Traffic Engineering*

The following **show** commands display information about IS-IS traffic engineering:

- **show isis mpls tunnel** – displays information about any tunnels that are used by IS-IS when calculating next hops. These are tunnels that are either registered with IS-IS when the MPLS tunnel is established or that are explicitly configured as static routes.
- **show isis database verbose** – displays MPLS traffic-engineering information about the IS-IS database.
- **show isis mpls advertisements** – displays the last record flooded from MPLS
- **show isis mpls adjacency-log** – displays a log of the last 20 IS-IS adjacency changes

For OSPF, you can use the **show ip ospf database opaque-area** command to display information about traffic-engineering opaque LSAs.

## MPLS and Differentiated Services

---

MPLS uses the EXP bits in the shim header to support differentiated services. You can set the EXP bits of outgoing MPLS traffic according to the internal traffic class and color of the traffic, assign internal traffic class and color to incoming MPLS traffic according to the EXP bits, or both.

For more information about the internal traffic class and color and support for differentiated services, see *ERX Policy and QoS Configuration Guide, Chapter 2, Configuring Quality of Service*.

### ***mpls match exp-bits***

- Use to set a combination of traffic class and color for incoming traffic matching the specified EXP bits value in the shim header.
- Specify an integer value in the range 0–7 to match the corresponding binary values (000–111) for the three EXP bits in the shim header.
- You can repeat the command to support the eight possible EXP bit values.
- Example

```
host1(config)#mpls match exp-bits 1 set traffic-class bronze  
color red
```

- The **no** version restores the default behavior, setting neither traffic class nor color for all traffic matching the specified EXP bits value.

**mpls match traffic-class**

- Use to set the EXP bits in the shim header of outgoing traffic matching a particular combination of traffic class and color.
- Specify an integer value in the range 0–7 to set the corresponding binary values (000–111) for the three EXP bits in the shim header.
- You can repeat the command to support up to 24 combinations: eight traffic classes supported on the system times three colors.
- Example
 

```
host1(config)#mpls match traffic-class traffic-class gold
                color green set exp-bits 7
```
- The **no** version restores the default behavior for traffic matching the specified traffic class and color. For matching traffic entering an LSP, it sets the EXP bits to 000. For matching transit traffic, it does nothing.

## Monitoring MPLS

---

Use the **show** commands in this section to monitor MPLS status.

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string you specify. Refer to *ERX System Basics Configuration Guide, Chapter 2, Command Line Interface*, for details.

You can trace paths through the MPLS user plane with the **traceroute** command. ICMP extensions enable LSRs to append MPLS header information (the label stack) to ICMP destination unreachable and time exceeded messages. The **traceroute** display shows the label and EXP bits used to switch the ICMP packets:

```
host1:edge1#traceroute 10.90.101.9
Tracing route to 10.90.101.9, TTL = 32, timeout = 2 sec.
(Press ^c to stop.)
 1  3ms  2ms  2ms      10.90.101.4    mplsLabel1=4009
    mplsExpBits1=0
 2  2ms  2ms  2ms      10.90.101.7    mplsLabel1=7004
    mplsExpBits1=0
 3  2ms  2ms  2ms      10.90.101.9
```

See *ERX Routing Protocols Configuration Guide, Vol. 1, Chapter 2, Configuring IP*, for more information on using the **traceroute** command.

**show cac**

- Use to display global CAC (call admission control) configuration.
- Example

```
host1#show cac
resource info flood interval 180
```

**show cac interface**

- Use to display all interfaces on which TE accounting is configured, or information only for the specified interface.
- Use the brief option to display summary information.
- Field descriptions
  - › bandwidth – maximum physical bandwidth in Kbps; line rate
  - › IP/MPLS reserveable bw – total bandwidth in Kbps that could be reserved for MPLS; includes bandwidth that is already reserved as well as bandwidth not yet reserved
  - › current total available bw – total bandwidth in Kbps that is available to be reserved
  - › MPLS TE flooding threshold up/down – absolute percentages of total reservable bandwidth that trigger the flooding of the new bandwidth value throughout the network; flooding is triggered when bandwidth increases past any of the up threshold values and when bandwidth decreases past any of the down threshold values
  - › MPLS TE administrative weight – weight assigned to the interface that supersedes a weight assigned by the IGP
  - › MPLS TE attribute flags – 32-bit value that assigns the interface to a resource class and enables a tunnel to discriminate among interfaces by matching against tunnel affinity bits
  - › Available BW at 8 priority levels – bandwidth in Kbps that is available at each priority level from 0–7
- Example

```

host1#show cac interface
atm2/0
  bandwidth 10 kbps
  IP/MPLS reserveable bw 10 kbps
  current total available bw 10 kbps
  MPLS TE flooding threshold:
    up   15 30 45 60 75 80 85 90 95 96 97 98 99 100
    down 100 99 98 97 96 95 90 85 80 75 60 45 30 15
  MPLS TE administrative weight 0
  MPLS TE attribute flags 0
  Available BW at 8 priority levels:
    0    10 kbps
    1    10 kbps
    2    10 kbps
    3    10 kbps
    4    10 kbps
    5    10 kbps
    6    10 kbps
    7    10 kbps

```

**show configuration**

- Use to display the configuration of all virtual routers or a specific virtual router.

- Example

```
host1#show configuration virtual-router euro7
```

### **show mpls**

- Use to display status and configuration information about MPLS.
- Field descriptions
  - › MPLS – status of MPLS, administratively enabled or disabled, and configuration status
  - › LSR ID – IP address of label-switched router
  - › Re-optimization timer – frequency at which LSPs are checked for better paths
  - › Label range – range of platform label space
  - › retry – retry behavior to be performed during LSP setup
  - › Loop Detect – status of loop detection, enabled or disabled
- Example

```
host1#show mpls
```

```
MPLS administratively enabled
Current state is Config incomplete
LSR ID is 2.2.2.2
Re-optimization timer is 3600
Label range 3000 ~ 4000
retry forever at interval 30 during LSP setup if there is route
retry forever at interval 30 during LSP setup if there is no route
Loop Detect enabled
```

### **show mpls binding**

- Displays label bindings for routes that are used for forwarding.
- Field descriptions
  - › in – label sent to upstream neighbor for displayed route
  - › out – label received from downstream neighbor for displayed route
  - › interface – interface on which the label is sent or received
  - › neighbor – IP address of neighbor to which the label is sent or received
- Example

```
host1#show mpls binding
```

```
Frame Relay over MPLS  vc-id 50001 group-id 2
  In   26 interface neighbor 222.9.1.3
  Out  27 interface neighbor 222.9.1.3
VLAN over MPLS  vc-id 240001 group-id 2
  In   22 interface neighbor 222.9.1.3
  Out  25 interface neighbor 222.9.1.3
```

```
10.1.1.1/32
```

```
  In   10001 interface ATM4/0.120 neighbor 10.4.12.2
```

```

Out    20001  interface ATM4/0.120  neighbor 10.4.12.2

10.2.2.2/32
In     10002  interface ATM4/0.120  neighbor 10.4.12.2
Out    20002  interface ATM4/0.120  neighbor 10.4.12.2

10.3.3.3/32
In     10005  interface ATM4/0.120  neighbor 10.4.12.2
Out    20003  interface ATM4/0.120  neighbor 10.4.12.2

10.4.12.0/30
In     10003  interface ATM4/0.120  neighbor 10.4.12.2
Out    20004  interface ATM4/0.120  neighbor 10.4.12.2

10.4.23.0/30
In     10004  interface ATM4/0.120  neighbor 10.4.12.2
Out    20005  interface ATM4/0.120  neighbor 10.4.12.2

```

### ***show mpls explicit-paths***

- Use to display all explicit paths or a particular explicit path.
- Field descriptions
  - › path name/identifier – name or identifier of explicit path and status, enabled or disabled, followed by list of path links and the IP address for each link's next address
- Examples

```

host1:pe2#show mpls explicit-paths
path name/identifier rx1-path enabled
1: next-address 70.70.70.2
2: next-address 30.30.30.1
not referenced by any options
path name/identifier rx1-path2 enabled
1: next-address 60.60.60.2
2: next-address 40.40.40.1
not referenced by any options

```

```

host1:pe2#show mpls explicit-paths name rx1-path2
path name/identifier rx1-path2 enabled
1: next-address 60.60.60.2
2: next-address 40.40.40.1
not referenced by any options

```

### ***show mpls forwarding***

- Use to display configuration and statistics for all LSPs or for specific LSPs configured on the LSR.
- Field descriptions

- › for – IP address and mask describing route for which LSP is created
  - › In label – label sent to upstream neighbor for route
  - › Out label – label received from downstream neighbor for route
  - › on – minor interface for LSP
  - › pkts – number of packets sent across LSP
  - › hcPkts – number of high-capacity (64-bit) packets sent across LSP
  - › octets – number of octets sent across LSP
  - › hcOctets – number of high-capacity (64-bit) octets sent across LSP
  - › errors – number of packets that are dropped for some reason before being sent
  - › discardPkts – number of packets that are discarded due to lack of buffer space before being sent
- Example

```
host1:vr2#show mpls forwarding
LSP tail-0a000200-30-42 for 10.0.2.0/255.255.255.252
  In label 2001 on atm2/1.120
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts

LSP tail-0a640202-32-43 for 10.100.2.2/255.255.255.255
  In label 2004 on atm2/1.120
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts

LSP tail-0a640203-32-45 for 10.100.2.3/255.255.255.255
  In label 2005 on atm2/1.120
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts

LSP to 10.100.3.3/255.255.255.255
  In label 2006 on atm2/1.120
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts
  Out label 3006 on atm2/0.230 to 10.0.2.2
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts

LSP to 10.100.3.4/255.255.255.255
  In label 2007 on atm2/1.120
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts
  Out label 3007 on atm2/0.230 to 10.0.2.2
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts
```

- Example

```
host1:vr2#show mpls forwarding brief
```

name/id	destination	label/intf
tail-0a000200-30-42	10.0.2.0/30	In 2001 on atm2/1.120
tail-0a640202-32-43	10.100.2.2/32	In 2004 on atm2/1.120
tail-0a640203-32-45	10.100.2.3/32	In 2005 on atm2/1.120
	10.100.3.3/32	In 2006 on atm2/1.120
		Out 3006 on atm2/0.230 to 10.0.2.2
	10.100.3.4/32	In 2007 on atm2/1.120
		Out 3007 on atm2/0.230 to 10.0.2.2

### **show mpls interface**

- Use to display status and configuration information about MPLS interfaces.
- Use the **shim** keyword to display information about shim interfaces. Use the **state not-up** keyword to display information about interfaces that are not in the up state.
- Field descriptions
  - › Interface – specifier and status of each interface
  - › RSVP – status of RSVP, configured or not, and profile used
  - › LDP/CR-LDP – status of LDP and CR-LDP, configured or not, and profile used
  - › IP interfaces on this MPLS interface – IP address of IP interfaces and session status
  - › Session statistics
    - notf – number of notification messages received or received bad or sent
    - mapping – number of label mapping messages received or received bad or sent
    - request – number of label request messages received or received bad or sent
    - release – number of label release messages received or received bad or sent
    - withdraw – number of label withdraw messages received or received bad or sent
    - addr – number of address messages received or received bad or sent
    - addr withdraw – number of address withdraw messages received or received bad or sent
  - › Adjacency statistics
    - hello rcv – number of hello messages received
    - hello sent – number of hello messages sent
    - bad hello rcv – number of hello messages received bad
    - adj setup time – time in *HH:MM:SS* since adjacency set up
    - last hello rcv time – time in *HH:MM:SS* since last hello message received
    - last hello sent time – time in *HH:MM:SS* since last hello message sent

› MPLS Statistics

- failed lbl lookup – number of packets received whose labels are not recognized
- octets – number of octets received or sent
- hcOctets – number of high-capacity (64-bit) octets received or sent
- pkts – number of packets received or sent
- hcPkts – number of high-capacity (64-bit) packets received or sent
- errors – number of packets that are dropped for some reason at receipt or before being sent
- discards – number of packets that are discarded due to lack of buffer space at receipt or before being sent

• Examples

```
host1:pe2#show mpls interface
```

```
Interface ATM2/0.120 Up
```

```
RSVP not configured
```

```
LDP/CR-LDP enabled with profile default
```

```
IP interfaces on this MPLS interface:
```

```
100.2.12.1/30
```

```
Session to 2.2.2.2 is operational (passive)
```

```
Session negotiated LDP advertisement mode is Downstream Unsolicited
```

```
Session statistics:
```

```
3 label alloc, 2 label learned, last reset flag = 0
```

```
last restart time = 00:00:19
```

```
Rcvd: 0 notf, 6 msg, 0 mapping, 0 request
```

```
0 abort, 0 release, 0 withdraw, 1 addr
```

```
0 addr withdraw, 6 msgId, 0 VCId msg
```

```
0 bad mapping, 0 bad request, 0 bad abort, 0 bad release
```

```
0 bad withdraw, 0 bad addr, 0 bad addr withdraw
```

```
0 bad VCId msg, 0 unknown msg type err
```

```
last info err code = 0, 0 loop detected
```

```
Sent: 0 notf, 6 msg, 3 mapping, 0 request
```

```
0 abort, 0 release, 0 withdraw, 1 addr
```

```
0 addr withdraw, 6 msgId, 0 VCId msg
```

```
Adjacency statistics:
```

```
5 hello rcv, 4 hello sent, 0 bad hello rcv, reset flag = 0
```

```
adj setup time = 00:00:19
```

```
last hello rcv time = 00:00:04, last hello sent time = 00:00:04
```

```
MPLS Statistics:
```

```
Rcvd: 0 failed lbl lookup, 0 octets, 0 hcOctets
```

```
0 pkts, 0 hcPkts, 0 errors, 0 discards
```

```
Sent: 0 octets, 0 hcOctets, 0 pkts
```

```
0 hcPkts, 0 errors, 0 discards
```

```
1 adjacency, 1 session
```

```
5 hello rcv, 5 hello sent, 0 hello rej
```

```
1 adj setup, 0 adj deleted, reset flag = 0
```

```
host1:pe2#show mpls interface
Interface atm2/0.60 Up
  RSVP not configured
  LDP/CR-LDP enabled with profile default
  IP interfaces on this MPLS interface:
    60.60.60.1/16 Session to 4.4.4.4 is operational
    (active)
  MPLS Statistics:
    Rcvd: 0 failed lbl lookup, 0 octets, 0 hcOctets
          0 pkts, 0 hcPkts, 0 errors, 0 discards
    Sent: 0 octets, 0 hcOctets, 0 pkts
          0 hcPkts, 0 errors, 0 discards
```

```
Interface atm2/0.70 Up
  RSVP not configured
  LDP/CR-LDP enabled with profile default
  IP interfaces on this MPLS interface:
    70.70.70.1/16
  MPLS Statistics:
    Rcvd: 0 failed lbl lookup, 0 octets, 0 hcOctets
          0 pkts, 0 hcPkts, 0 errors, 0 discards
    Sent: 0 octets, 0 hcOctets, 0 pkts
          0 hcPkts, 0 errors, 0 discards
```

```
host1:pe2#show mpls interface atm 2/0.60
Interface atm2/0.60 Up
  RSVP not configured
  LDP/CR-LDP enabled with profile default
  IP interfaces on this MPLS interface:
    60.60.60.1/16 Session to 4.4.4.4 is operational
    (active)
  MPLS Statistics:
    Rcvd: 0 failed lbl lookup, 0 octets, 0 hcOctets
          0 pkts, 0 hcPkts, 0 errors, 0 discards
    Sent: 0 octets, 0 hcOctets, 0 pkts
          0 hcPkts, 0 errors, 0 discards
```

```
host1:pe2#show mpls interface brief
Interface      IP-Address      Protocol  Status
atm2/0.60      60.60.60.1/16  ldp      Up
atm2/0.70      70.70.70.1/16  ldp      Up
```

### ***show mpls ldp targeted session***

- Use to display LDP targeted hello receive or send list, or both.
- Field descriptions

- › D – targeted session created by layer 2 over MPLS connection
- › S – targeted session statically created by user
- › A – targeted session created by access list
- › Used By – letter representing source of targeted session
- Example

```
host1#show mpls ldp targeted session
Mpls Target Session Status:
```

D = Dynamically, S = Statically, A = Access List Configured

```
Targeted session sent to 6.6.6.6 is Up Used By: A
Targeted session sent to 7.7.7.7 is Up Used By: A
Targeted session sent to 8.8.8.8 is Up Used By: A
Targeted session sent to 9.9.9.9 is Up Used By: S/A
Targeted session sent to 10.10.10.10 is Up Used By: A
Targeted session sent to 2.2.2.2 is Down Used By: S
Targeted session received from 3.3.3.3 is Down Used By: S
```

### ***show mpls minor-interface***

- Use to for display all minor interfaces (including those for tunnels) or a specific minor interface in the current router context.
- Field descriptions
  - › Tunnel – tunnel name
  - › State – status of tunnel, up or down
  - › LSP setup using – label distribution protocol used to establish tunnel
  - › In, Out Label – label prepended to packets before being sent across tunnel
  - › Residing on – location of tunnel
  - › tunnel is announced to – protocols to which the tunnel is announced
  - › metric – metric type, relative or absolute
  - › pkts – number of packets sent across tunnel
  - › hcpkts – number of high-capacity (64-bit) packets sent across tunnel
  - › octets – number of octets sent across tunnel
  - › hcocets – number of high-capacity (64-bit) octets sent across tunnel
  - › errors – number of packets that are dropped for some reason before being sent
  - › discardPkts – number of packets that are discarded due to lack of buffer space before being sent
- Example

```
host1:two#show mpls minor-interface
```

```
Tunnel 2 to 0.0.0.0
State: Enabled with Incomplete Config
LSP setup using cr-ldp
```

```
tunnel not announced to any IGP
(Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is route
(Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is no route
metric is relative 0
```

```
Tunnel tail-de090106-1-18a for 222.9.1.2
```

```
State: Up
In label 20 on ATM3/0.1
    0 pkts, 0 hcPkts, 0 octets
    0 hcOctets, 0 errors, 0 discardPkts
```

```
FastEthernet2/4.1 to 222.9.1.3
```

```
State: Up
In label 25
    14 pkts, 0 hcPkts, 2100 octets
    0 hcOctets, 0 errors, 0 discardPkts
Out label 18 on tun mpls:1 nbr 222.9.1.3
14 pkts, 0 hcPkts, 2100 octets
    0 hcOctets, 0 errors, 0 discardPkts
```

```
FastEthernet2/4.1 to 222.9.1.3
```

```
State: Up
```

```
FastEthernet2/4.1 to 222.9.1.3
```

```
State: Up
Out label 18 on tun mpls:1 nbr 222.9.1.3
    14 pkts, 0 hcPkts, 2100 octets
    0 hcOctets, 0 errors, 0 discardPkts
```

### ***show mpls profile***

- Use to display a specific LDP, RSVP-TE, or tunnel profile, or all LDP, RSVP-TE, or tunnel profiles.
- Field descriptions
  - › profile – number of interfaces that use the profile
  - › session retry – number of attempts that will be made to set up an MPLS LDP session
  - › CR-LDP – status of CR-LDP traffic-engineering extensions to LDP
  - › refresh-period – timeout period in seconds between generation of refresh messages
  - › timeout factor – number of refresh messages that can be lost before the session is ended
- Examples

```
host1:pe2#show mpls ldp profile default
ldp profile default: used by 2 interfaces
  session retry: 10 times at interval 10
  CR-LDP enabled
```

```
host1:pe2#show mpls rsvp profile default
RSVP profile default: used by 0 interfaces
  refresh period: 30000 ms
  timeout factor: 3
```

### ***show mpls rsvp***

- Use to display RSVP path state control blocks, reservation state control blocks, or session information on the virtual router to assist in debugging.
- Sessions can be any of the following:
  - › ingress – originating on the router
  - › egress – terminating on the router
  - › transit – travelling through the router
- Field descriptions
  - › PSB – path state control block
  - › RSB – reservation state control block
  - › Sender – IP address of PSB or RSB sender
  - › LSPId – ID of LSP
  - › timeout – amount of time before PSB/RSB times out if no refresh arrives.
  - › InLabel – incoming label information
  - › Associated Minor Interface – tunnel identifier for minor interface for which the RSVP information is displayed
  - › pHopIntf – penultimate hop interface
  - › IncomingIntf – incoming interface
  - › OutgoingIntf – outgoing interface
  - › PHopAddr – penultimate hop address

- › m\_ipNextHopAddr – next hop address
- › NextHop – type of next hop (loose, strict, session)
- › LabelRange – RSVP session label range
- › SenderTSpec – traffic parameters for the sender
  - Token Bucket Rate – sender's description of generated traffic
  - Token Bucket Size – sender's description of generated traffic
  - Peak Data Rate – sender's peak traffic generation rate
  - Min Policed Unit – minimum packet size generated by sender
  - Max Packet Size – maximum packet size generated by sender
- › RRO – record route object
- › ADSPEC – indicates presence of this QoS object
- › IN ERO – incoming explicit route object
- › OUT ERO – outgoing explicit route object
- › SES ATTR – RSVP session attributes
  - Setup Pri – setup priority of tunnel
  - Hold Pri – hold priority of tunnel
  - name – name of the tunnel
  - Flags – one or more of the IngressReRoute (the ingress router can reroute the LSP), Local Protection (routers can use local repair mechanism to fix the LSP; this fix might violate the explicit route object associated with the LSP), and MergingPermitted (LSPs can be merged) flags
- › TTC – indicates presence of the traffic trunk classifier object
- › Policy Object – indicates presence of the policy object
- › Unknown Objects – indicates presence objects not defined by the RSVP specification
- › PSB Flags
  - InUse – PSB is in use
  - Deleted – PSB is deleted
  - NonRsvp – non-RSVP hop present
  - RouteChangeNotify – route change notification received
  - EroChanged – explicit route object changed
  - NextHopChanged – next hop has changed
  - RtNextHopChanged – routing table next hop changed
  - EgressStatusChanged – PSB egress status has changed
  - QosChanged – QoS characteristics have changed
  - LabelChanged – label has changed
  - ResvRefreshNeeded – reservation refresh needed
  - PathRefreshNeeded – path refresh needed
  - RroRequired – record route object required
  - Egress – session is egress
  - PathRefreshSent – path refresh sent

- EgressFilterFF – egress filter reservation style Fixed Filter
- QosCorrectionNeeded – QoS correction needed
- IsPathTrigger – has path refresh been triggered
- › RSB Flags
  - InUse – is RSB in use
  - Deleted – has RSB been deleted
  - RcvdAck – has ack been received
  - StyleConverted – has reservation style been converted to shared explicit
  - IsPathTrigger – has reservation refresh been triggered
- › Destination – RSVP session destination address
- › TunnelId – RSVP session tunnel ID
- › Extended Tunnel Id – RSVP session extended tunnel ID
- Examples for an ingress session

```
host1#show mpls rsvp psb
```

```
PSB: Sender 223.10.1.1 LSPId 1 timeout -- InLabel --
     PHopIntf
     IncomingIntf
     OutgoingIntf ATM2/0.1
     PHopAddr 0.0.0.0 m_ipNextHopAddr 221.1.1.1
     NextHop 221.1.1.1/255.255.255.255 (strict)
     LabelRange --
     SenderTSpec CType IntServ Controlled Load
                Token Bucket Rate 0
                Token Bucket Size 0
                Peak Data Rate 0
                Min Policed Unit 0
                Max Packet Size 0
     Flags : InUse
           RroRequired
           PathRefreshSent
```

```
host1#show mpls rsvp rsb
```

```
RSB: Timeout 157500 label 1/33
     Flags : InUse
           StyleConverted
```

```
host1:two#show mpls rsvp sessions
```

```
Destination 222.9.3.1 TunnelId 1 Extended Tunnel Id
223.10.1.1
PSB: Sender 223.10.1.1 LSPId 1 timeout 157500 InLabel 17
     Associated Minor Interface: Tunnel 223.10.1.1:1
     PHopIntf ATM2/0.1
     IncomingIntf ATM2/1.1
     OutgoingIntf ATM2/0.3
     PHopAddr 221.1.1.2 m_ipNextHopAddr 122.1.1.1
```

```

NextHop 122.1.1.1/255.255.255.255 (strict)
LabelRange (generic) min 0 max 1048575
SenderTSpec CType IntServ Controlled Load
    Token Bucket Rate 0
    Token Bucket Size 0
    Peak Data Rate 0
    Min Policed Unit 0
    Max Packet Size 0
RRO IPv4 hop 221.1.1.2 (strict)
ADSPEC --
IN ERO IPv4 hop 221.1.1.1 (strict)
    IPv4 hop 122.1.1.1 (strict)
OUT ERO IPv4 hop 122.1.1.1 (strict)
SES ATTR Setup Pri 4, Hold Pri 4, name --
    Flags : IngressReRoute
TTC --
Policy Object --
Unknown Objects --
Flags : InUse
    PathRefreshSent
RSB: Timeout 157500 label 16
Associated Minor Interface: Tunnel 223.10.1.1:1
FlowSpec CType IntServ Controlled Load
    Token Bucket Rate 0
    Token Bucket Size 0
    Peak Data Rate 0
    Min Policed Unit 0
    Max Packet Size 0
RRO IPv4 hop 122.1.1.1 (strict)
Policy Object --
Unknown Objects --
Flags : InUse
    StyleConverted
    ResvRefrSent

```

### ***show mpls tunnels***

- Use to display status and configuration for all tunnels or for a specific tunnel in the current router context.
- A result of Incomplete Configuration in the display indicates either no tunnel endpoint or no label distribution protocol.
- Field descriptions
  - › Label – label prepended to packets before being sent across tunnel
  - › State – status of tunnel, up or down
  - › Residing on – location of tunnel
  - › LSP setup using – label distribution protocol used to establish tunnel

- › tunnel is announced to – protocols to which the tunnel is announced
- › metric – metric type, relative or absolute
- › Mpls Statistics
  - pkts – number of packets sent across tunnel
  - hcPkts – number of high-capacity (64-bit) packets sent across tunnel
  - octets – number of octets sent across tunnel
  - hcOctets – number of high-capacity (64-bit) octets sent across tunnel
  - errors – number of packets that are dropped for some reason before being sent
  - discardPkts – number of packets that are discarded due to lack of buffer space before being sent

- Examples

```
host1:pe2#show mpls tunnels
```

```
Tunnel 2 to 0.0.0.0
  State: Enabled with Incomplete Config
  LSP setup using cr-ldp
  tunnel not announced to any IGP
  (Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is route
  (Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is no route
  metric is relative 0

Tunnel 1 to 222.9.1.3
  State: Up
  Out label 24 on ATM3/0.1 nbr 10.10.11.5
    14 pkts, 0 hcPkts, 2156 octets
    0 hcOctets, 0 errors, 0 discardPkts
  LSP setup using cr-ldp
  tunnel not announced to any IGP
  (Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is route
  (Global) Retry forever
    at (Global) interval 5 during Lsp setup if there is no route
  metric is relative 0
  path option 2
    option is currently used - path is calculated by isis
      10.10.11.5
      10.10.12.3
      222.9.1.3
    next reoptimization in 1687 seconds
  stacked labels:
FastEthernet2/4.1 222.9.1.3 R0 Out 18 on tun mpls:1
```

```
Tunnel tail-de090106-1-18a for 222.9.1.2
  State: Up
  In label 20 on ATM3/0.1
    0 pkts, 0 hcPkts, 0 octets
0 hcOctets, 0 errors, 0 discardPkts
```

```
host1:pe2#show mpls tunnels role head
```

```
Tunnel pe2-to-pe1 to 1.1.1.1
  Out Label 300
  State: Up
  Residing on atm2/0.60
  LSP setup using ldp
  tunnel is announced to ISIS BGP
  metric is relative 0
```

```
Mpls Statistics:
  0 pkts, 0 hcPkts, 0 octets
  0 hcOctets, 0 errors, 0 discardPkts
```

```
host1:pe2#show mpls tunnels role tail
```

```
Tunnel for 2.2.2.2
  In Label 3000
  State: Up
  Residing on atm2/0.70
```

```
Mpls Statistics:
  0 pkts, 0 hcPkts, 0 octets
  0 hcOctets, 0 errors, 0 discardPkts
```

```
host1:pe2#show mpls tunnels destination 2.2.2.2
```

```
Tunnel for 2.2.2.2
  In Label 3000
  State: Up
  Residing on atm2/0.70
```

```
Mpls Statistics:
  0 pkts, 0 hcPkts, 0 octets
  0 hcOctets, 0 errors, 0 discardPkts
```

**show mpls tunnels brief**

- Use the **brief** keyword to display a summary of all MPLS tunnels for the current router context or summary information for a specific tunnel in the current router context.
- Field descriptions
  - › name/id – tunnel identifier
  - › destination – tunnel destination; router ID of egress router
  - › metric – value of tunnel metric, relative or absolute
  - › state/label/intf – functional state of tunnel, label prepended to packets sent across tunnel, and interface where tunnel resides
- Example

```
host1:pe2#show mpls tunnels brief
name/id          destination  metric  state/label/intf
vpnEgressLabel3  0.0.0.0     R0      Incoming 1048573 on stack
vpnEgressLabel4  0.0.0.0     R0      Incoming 1048572 on stack
pe2-to-pe1       1.1.1.1     R0      Outgoing 300 on atm2/0.60
                  2.2.2.2     R0      Incoming 3000 on atm2/0.70
```