MPLS OAM Tools for Troubleshooting MPLS Networks

Increasing number of networks are migrating to Multiprotocol Label Switching (MPLS) because of the tremendous benefits it offers. At the same time, the separation of data and the control plane in MPLS introduces additional complexity in troubleshooting MPLS networks.

This document includes the following topics:

- Operation of the traditional ping and trace troubleshooting methods in the MPLS environment.
- Operation of the new MPLS operations, administration, and maintenance (OAM) capabilities of Label Switched Path (LSP) Ping and LSP Traceroute.
- Case studies of using LSP Ping/Traceroute in troubleshooting scenarios

Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Contents

Understanding Traditional Ping and Trace

The traditional ping uses an Internet Control Message Protocol (ICMP) packet generated as an echo request (ICMP type 8). The source router emits the packet for the destination specified by the **ping** command. If no source address is given (through extended ping), the router uses the outgoing interface towards the next hop. Upon successfully reaching the destination, the IP host replies with an ICMP echo reply (type 0), which is routed back towards the IP address used to source the packet.

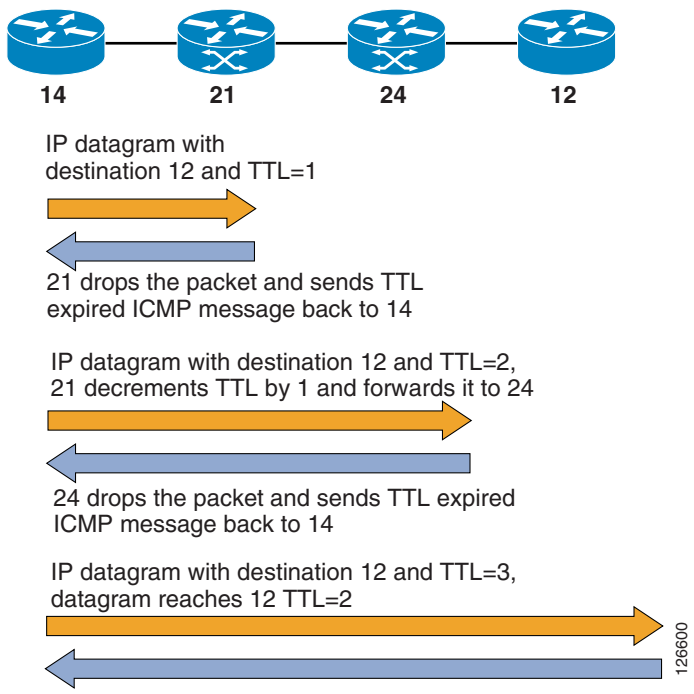
If the packet encounters maximum transmission unit (MTU), time-to-live (TTL), or other routing issues along the path between the originating router and the target destination, the request times out because by design (RFC), intermediate routers do not generate an ICMP packet to notify a sender about errors occurring on an ICMP packet. The result displays five ping attempts and the time for the source to receive the answer from the destination.

The traditional trace uses a UDP packet sent with incremental TTL values starting from 1 and up by default. Upon receiving a packet with TTL=1, a router process-switches the packet and emits an ICMP timeout (ICMP type 11) towards the source router. This means that along the way between the source and destination, each router successively receives a packet with a TTL of 1 and replies to the source, thus giving information about the path of routers between the source and destination.

Upon reaching the destination, the destination host or router replies with an ICMP packet of destination unreachable and port unreachable (type 3 and code 3).

Figure 1 shows an example of this operation.

Figure 1 Traditional Ping and Trace Operation



Ping and Trace Encapsulation for MPLS

For MPLS networks, enhancements have been made to ICMP to support MPLS encapsulation. For a ping operation, the ICMP packet is sent as IP, or with an MPLS packet if a Forwarding Equivalence Class (FEC) exists on the source router for this IP destination. The same conditions apply for the ICMP on the return path. Note that, depending on symmetric or asymmetric routing and on load sharing in the MPLS cloud, the ICMP response packets might not use the same path, and in any case use different labels (an LSP is one way).

For trace operation, the UDP packet is encapsulated in the same way. When the TTL expires, the intermediate router sends an ICMP timeout packet (type 11) back to the emitting source. This packet is similarly encapsulated if an FEC is available.

When an ICMP packet is generated in response to a received MPLS-encapsulated packet, by definition the ICMP response should contain the original MPLS label stack, including information such as S, EXP, TTL, and so on, as part of the ICMP payload information.

Note For more details on ICMP extensions, see "draft-ietf-mpls-icmp-02.txt" at the following URL: http://www.ietf.org/proceedings/00dec/I-D/draft-ietf-mpls-icmp-02.txt

This feature enables the trace for MPLS networks to display the label stack received from each middle router (Label Switch Router [LSR]), located on the path between the source and destination. The trace results also display each incoming interface of the routers along the path.

The MPLS extensions for ICMP also specify that an ICMP packet sent by an LSR in response to an unreachable IP packet must be able to reach the packet IP source, including in the case of a source private address or if the LSR is part of an MPLS transit network.

Note For more details on MPLS label encapsulation, see "draft-ietf-mpls-label-encaps-08.txt" at the following URL: http://www.potaroo.net/ietf/all-ids/draft-ietf-mpls-label-encaps-08.txt-47028.txt

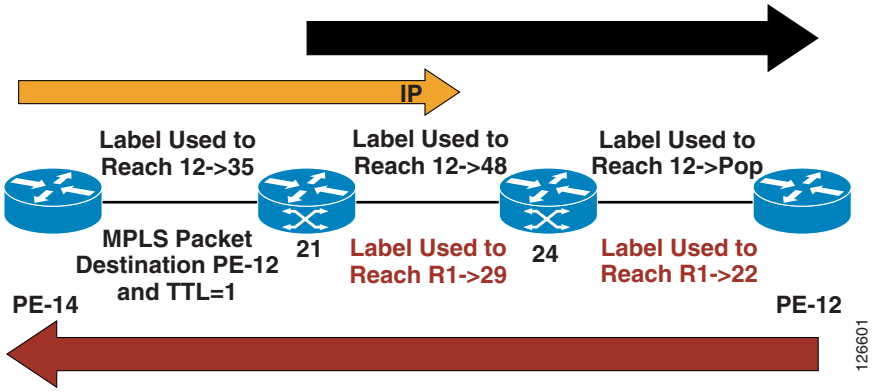
The Cisco implementation reapplies the same MPLS stack to the ICMP response packet and changes the TTL value of all the labels to 255. The label stack corresponds to the same FEC as the one for which the packet was received. The ICMP response packet is then sent all the way on the LSP for the initial MPLS packet destination. Upon receiving this ICMP response, the last LSR of the LSP performs an IP lookup on the IP destination address and forwards the ICMP response towards the initial source.

Note Note that for this reason, the round trip time information obtained from a traceroute in an MPLS network is not valid. See TTL Hiding, page 5 for details about the **mpls ip ttl-expiration pop** command option for an exception to this.

If a failure occurs along the LSP path between the source and destination because of a problem such as MTU restrictions, a middle LSR emits an ICMP packet. This packet follows the LSP all the way and is then sent back to the source. If the LSP path is broken, the ICMP packets sent out by the middle LSRs with the copied label stack might never reach the source.

Figure 2 shows an example of ping and trace encapsulation for MPLS.

Figure 2 Ping and Trace Encapsulation for MPLS Example



In the following configuration example, the loopback address of PE-12 is pinged. MPLS is enabled between the provider edge (PE) routers and the Ps.

```
pop1-7206-14-PE#trace 135.15.252.12

Type escape sequence to abort.
Tracing the route to 135.15.252.12

  1 135.15.210.21 [MPLS: Label 35 Exp 0] 0 msec 4 msec 0 msec
  2 135.15.126.24 [MPLS: Label 48 Exp 0] 0 msec 0 msec 0 msec
  3 135.15.230.12 0 msec *  0 msec
```

Note that this example defines the following:

- An LSR only emit s an ICMP message if the underlying protocol of the received MPLS packet is IP.
- No ICMP message is generated in response to a failure to deliver a received ICMP message. Therefore, if an ICMP echo message (ping sent as IP or MPLS) times out or, for example, it cannot be fragmented, no ICMP message is sent to notify the source.

TTL Hiding

The **mpls ip propagate-ttl** [*local* | *forwarded*] command enables a router to manipulate the TTL for a packet that has been either generated locally or has been forwarded by the router. Both options are possible and configurable. When this feature is enabled, a traceroute that uses an LSP to reach an FEC destination might not allow for the hop-by-hop path to be revealed, depending on which option is being used. For example, this allows the core routers of a network to be hidden from customers but not from a service provider (SP) operations team.

The **mpls ip propagate-ttl** [*local* | *forwarded*] command allows to control whether the label stack corresponding to the original LSP is used to forward the packet, or if the packet is forwarded based on the original source IP information (which is set as the ICMP reply packet destination).

Depending on the number of labels on a received initial packet, the [**no**] **mpls ip ttl-expiration pop** *min#labels* command allows to choose whether or not to use the MPLS extensions for ICMP (see Ping and Trace Encapsulation for MPLS, page 4), in relation to the number of labels contained by the incoming packet label stack.For example, if an MPLS packet received with the same or a lower number of labels than the *min#labels* needs ICMP processing because of unreachability, the ICMP response

packet is forwarded back towards the IP source of the initial packet using the IP routing information from the global or the VRF routing table (if the incoming label is associated with a VRF). MPLS packets with higher number of labels use the MPLS extensions implementation for ICMP.

Support of VRF Aware Ping and Trace using MPLS Encapsulation

VPN Routing and Forwarding (VRF) Aware Ping and Traceroute differ from the traditional ping and trace by using the VRF routing table for route lookup instead of the global routing table when sending the probe packets. This is used primarily for private address spaces.

If no source IP address is given, the router uses the first interface that is associated with the VRF and is up. If you choose a source IP address using the extended command, ensure that the corresponding interface is up and is associated with the VRF; otherwise, the destination might not be able to route the response back.

The VRF-associated interfaces are shown by the **sh ip vrf int** *vrf_name* command. Note that the protocol status is shown as well.

```
pop1-7206-14-PE#sh ip vrf int SAA
Interface              IP-Address      VRF                             Protocol
FastEthernet2/0        135.15.1.14     SAA                             up
Loopback3              3.3.3.3         SAA                             up
```

The way VRF routing works, the core routers between the PEs do not know how to reach the source and destination routers. Thus, the core routers are not able to respond directly to the VRF interface that sourced the UDP packets.

The implementation of VRF Aware Ping does not rely on a response from the middle LSRs and thus does not present any routing issues. If a failure occurs along the path between the PEs, the probe ICMP packet is lost (no response reaches the source), because no ICMP packet may be generated in response to an error on an ICMP packet.

The implementation of VRF Aware Traceroute relies on mechanisms defined by draft-ietf-mpls-label-encaps-08.txt. ICMP replies that are generated when TTL expires or when the destination is unreachable are sent in the same direction as the initial packet. Upon reaching the egress PE router, the VPN label is popped and routing is performed using the VRF information based on the IP header of the ICMP response packet. The response ICMP packet is then sent in the direction of the source, potentially using MPLS encapsulation as well.

Following the usual trace mechanism, additional UDP packets are sent to the destination with incremental TTL values. These packets will expire along the LSP path and the corresponding core routers will handle the expired packets similarly as explained above. The trace results displayed on the source router show the label stack information and the IP address of each incoming interface of the LSR routers along the LSP, as shown in the example below, where 135.15.3.73 is the IP address of a remote CE.

```
pop1-7206-14-PE#sh ip route vrf SAA

Routing Table: SAA
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR

Gateway of last resort is not set
```

```
     51.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
B       51.1.1.51/32 is directly connected, 16:31:59, Serial4/0
B       51.1.1.0/24 is directly connected, 16:31:59, Serial4/0
     135.15.0.0/16 is variably subnetted, 7 subnets, 2 masks
S       135.15.252.71/32 [1/0] via 135.15.1.71, FastEthernet2/0
B       135.15.252.72/32 [200/0] via 135.15.252.11, 07:52:24
B       135.15.252.73/32 [200/0] via 135.15.252.12, 07:52:24
B       135.15.249.51/32 [20/0] via 51.1.1.51 (vrf1), 16:31:59, Serial4/0
C       135.15.1.0/24 is directly connected, FastEthernet2/0
B       135.15.2.0/24 [200/0] via 135.15.252.11, 07:52:24
B       135.15.3.0/24 [200/0] via 135.15.252.12, 07:52:24
```

```
pop1-7206-14-PE#trace vrf SAA 135.15.252.73

Type escape sequence to abort.
Tracing the route to 135.15.252.73

  1 135.15.210.21 [MPLS: Labels 35/22 Exp 0] 0 msec 0 msec 0 msec
  2 135.15.126.24 [MPLS: Labels 48/22 Exp 0] 4 msec 0 msec 0 msec
  3 135.15.3.12 [MPLS: Label 22 Exp 0] 0 msec 0 msec 0 msec
  4 135.15.3.73 4 msec *  0 msec
pop1-7206-14-PE#
```

Note the following about the above configuration example:

- A two-label stack is displayed; the VPN label is 22 and the top Label Distribution Protocol (LDP) labels are 35 and 48 for the LSP going from PE-14 to PE-12.
- The interface shown on the egress PE is the VRF interface towards the customer edge (CE), not the core incoming interface (facing the last P router) as one might expect (note that incoming interfaces are displayed for each router along the path).
- The VPN label is displayed only on the PE line; the last P router performed penultimate hop popping (PHP) and the egress PE received a packet with a VPN label in the stack only. The egress PE router is not the final destination; therefore, the label stack containing the VPN label is sent as the payload of the ICMP timeout reply towards the emitting source.

In the following example, the destination IP address is a VRF interface on the egress PE itself:

```
pop1-7206-14-PE#trace vrf SAA 135.15.3.12

Type escape sequence to abort.
Tracing the route to 135.15.3.12

  1 135.15.210.21 [MPLS: Labels 35/23 Exp 0] 0 msec 0 msec 0 msec
  2 135.15.126.24 [MPLS: Labels 48/23 Exp 0] 4 msec 0 msec 0 msec
  3 135.15.3.12 0 msec *  0 msec
pop1-7206-14-PE#
```

Note that in the above example, the VPN label is not displayed on the third line. The PE that is the final destination of the packet receives a label packet after PHP. The Label is a VPN label, which is popped and used to identify the VRF routing table to use for IP routing lookup. At this point the packet is an IP packet handled by the UDP ICMP code, which does not take into account the label stack. The router will thus process the TTL expired UDP packet and emit an ICMP response packet with no label stack in its payload.

LSP Ping/Traceroute Operation

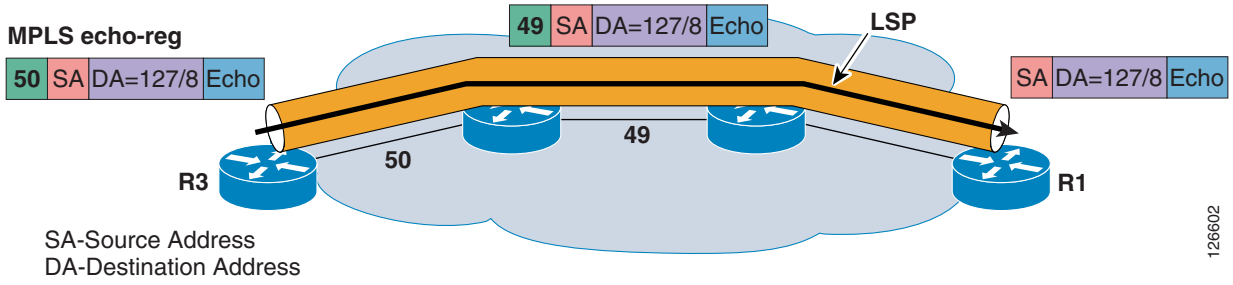
When an LSP fails to deliver the data traffic, the LSP failure cannot always be detected using the traditional ICMP ping and traceroute. The LSP Ping/Traceroute OAM tools provide additional methods for LSP failure detection and diagnosis by enabling users to quickly detect traffic "blackholes" or misrouting and thus isolate faults because of LSP failure.

Similar to the traditional IP ping, LSP Ping/Traceroute is based on echo request and echo reply, but does not use an ICMP packet. LSP Ping/Traceroute relies on IPv4 or IPv6 UDP packets with port 3503. UDP packets received on this port are either an MPLS echo or an MPLS echo reply. MPLS LSP echo request/reply is useful in testing an LSP for a specific FEC stack. For the LSP ping/trace packet to be treated as a control packet, the same label stack is used as is used by the LSP, which causes the echo to be switched inband of the LSP tested. The destination of the echo request is a 127.x.y.z/8 address that is not the same as the one used for selecting the LSP.

Any router using the127/8 address for forwarding (because of a broken LSP) punts the packet for local processing. 127/8 also forces the packet to be processed by the route processor (RP) at the egress router for a given LSP. If the LSP is set up using LDP and is mapped to an egress IP address of 10.1.1.1, the FEC stack contains a single element, which is an LDP IPv4 prefix sub-Type-Length-Value (TLV) with value 10.1.1.1/32. If the LSP being tested is a Resource Reservation Protocol (RSVP) LSP, the FEC stack consists of a single element that captures the RSVP session and sender template that uniquely identifies the LSP.

In Figure 3, LSP Ping/Traceroute uses the same label stack as is used by the LSP, which causes the echo to be switched inband of LSP. The IP header destination address field of the echo request is a 127/8 address.

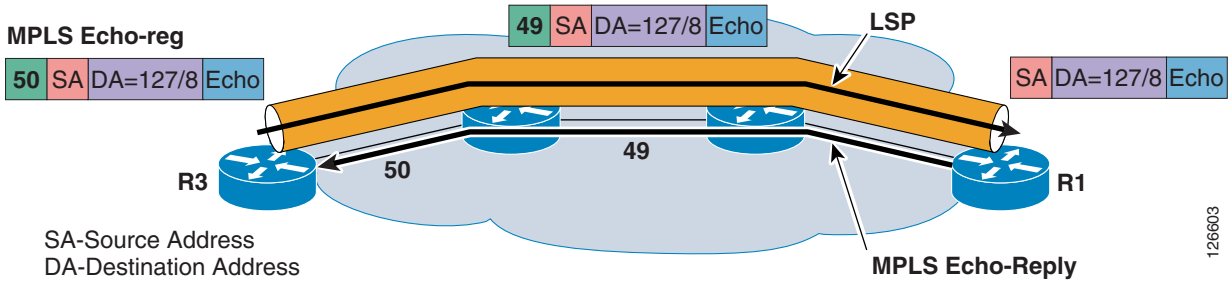
Figure 3 MPLS Echo Request



An echo reply, which may or not be labeled, has the outgoing interface IP address as the source. The destination IP address and port are copied from the source address and port of the echo request. The echo reply can thus take an MPLS or an IP path to return to the source router. LSP Ping/Traceroute verifies only the LSP paths in one direction and not the LSP return path.

Figure 4 shows that the return path for the echo reply can be same or different from the path taken by the request.

Figure 4 Echo Reply Return Path

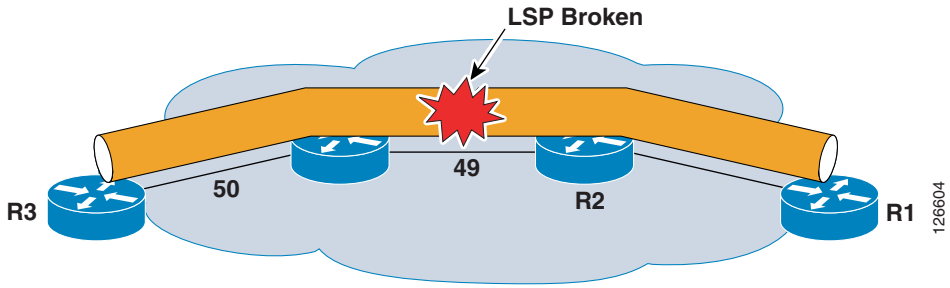


There can be various reasons for the LSP to break, including the following:

- Broken LDP adjacency
- MPLS not enabled
- Mismatch labels
- Software/hardware corruption

As Figure 5 shows, a normal IP ping is successful when an LSP is broken if there is still IP connectivity.

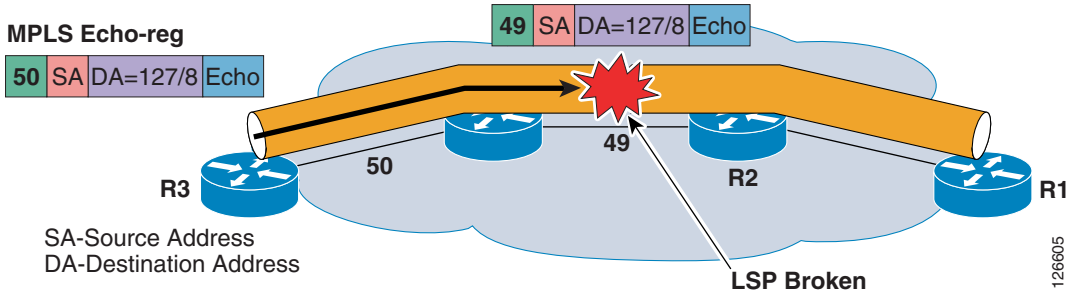
Figure 5 LSP Broken Path



The presence of the 127/8 address in the IP header destination address field causes the packet to be consumed by any router trying to forward the packet using the IP header. In this case, R2 does not forward the echo request to R1 but rather consumes the packet and replies to it accordingly.

As Figure 6 shows, an LSP ping packet is consumed by the router that has a broken LSP.

Figure 6 LSP Packet Lost because of Broken LSP



The operation of an LSP ping is based on examining the FEC whole MPLS path, which is supposed to be checked. The echo request is forwarded just like any other packet belonging to that FEC.

In LSP ping mode, a connectivity check is performed for an MPLS network. The echo packet reaches the destination router, which then processes the packet, verifying that it is indeed an egress for the FEC.

In LSP traceroute mode, the packet is processed at each transit LSR, which performs various checks to ensure that it is indeed a transit LSR for this path. This LSR also returns further information that helps check the control plane against the data plane; that is, checking that the forwarding matches what the routing protocols have determined as the path.

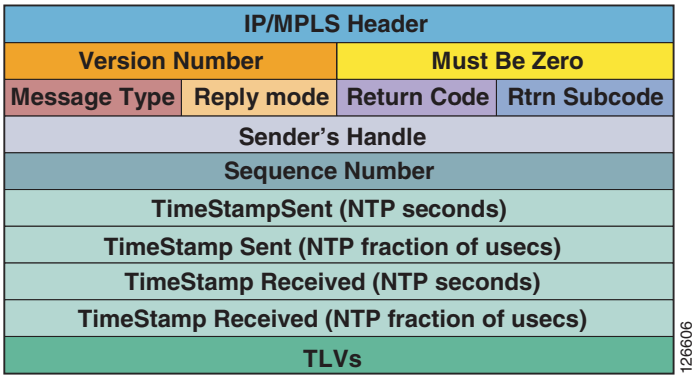
LSP Echo Request and Reply Packet Format

LSP echo mode is used to test the integrity of connectivity using the verification on the FEC entity between the ping origin and the egress node for this particular FEC. This test is carried out by sending an MPLS echo request along the same data path as other packets belonging to this FEC, as shown in Figure 1. As such, it is forwarded like any packet belonging to that FEC. The MPLS echo request contains information in its payload about the FEC whose MPLS path is being verified.

Once the MPLS request packet reaches the end of the path, the egress LSR verifies that it is an egress for the FEC and notifies the node that originated the MPLS echo request with the appropriate return code.

The packet format of the LSP ping is a UDP packet with the fields shown in Figure 7.

Figure 7 Packet Format of LSP Ping



Version Number

The Version Number field contains the version of the MPLS echo packet. It is currently set to 1.

Message Type

The Message Type field specifies whether the packet is an MPLS echo request or an MPLS echo reply, as shown in Table 1.

Table 1 MPLS Echo Request and Reply Format

Field Number	Field Value
1	MPLS Echo Request
2	MPLS Echo Reply

Reply Mode

The Reply Mode field is used to control how the target router replies to MPLS echo request, as shown in Table 2.

Table 2 Reply Mode

Field Number	Field Value	Description
1	Do not reply	• This option is used to verify the one way path connectivity. The receiving router does not need to reply to the ping message. The receiving router may log gaps in the sequence numbers and/or maintain delay/jitter statistics. • Cisco IOS Release 12.0(27)S does not use this mode, but Cisco does support it. • This mode is useful for a keepalive application running at the remote end. Such an application triggers state changes if it does not receive an LSP ping packet with a predefined time. • An MPLS echo request with "Do not reply" may also be used by the receiving router to log gaps in the sequence numbers and/or maintain delay/jitter statistics.
2	Rely via an IPv4 UDP packet	• This implies that an IPv4 UDP packet should be sent in reply to an MPLS echo request. • This is the most common reply mode for simple MPLS LSP pings sent to periodically poll the integrity of an LSP. • This is the default reply mode.
3	Reply via an IPv6 UDP packet with router alert	• In this mode, when the destination router replies, it appends a label of "1" to the packet. • This forces all the intermediate routers, on the way back, to process-switch the reply. • This mode is CPU-intensive and should generally be used if the reply fails for "reply with IPv4 UDP packet". • This mode is useful when there is an inconsistency between IP and MPLS.

Return Code

The router initiating the LSP ping/traceroute sets the return code to zero. The replying router sets it accordingly, based on the table in Table 3.

Table 3 Return Codes

Field Number	Field Value
0	The error code is contained in the error code TLV
1	Malformed echo request received
2	One or more of the TLVs was not understood
3	Replying router is an egress for the FEC
4	Replying router has no mapping for the FEC
5	Replying router is not one of the "downstream routers"
6	Replying router is one of the "downstream routers," and its mapping for this FEC on the received interface is the given label

Sender Handle

The Sender Handle field is added by the sender in the echo request. The recipient puts the same value back in the echo reply. The sender handle is relevant only to the sender, which uses the value to match the echo reply against the echo request. The value remains the same for all counts of a single ping.

The following debug example shows the sender handle information when doing an LSP ping and turning on the LSPV debug. The following debugs are taken from a router receiving an LSP ping:

```
*Jan  9 10:12:26.495: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle F60000B5, sequence number 1, ti
mestamp sent 09:59:07 UTC Fri Jan 9 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
1900
*Jan  9 10:12:26.495: LSPV: Echo Hdr encode: version 1, msg type 2, reply mode 2
, return_code 3, return_subcode 0, sender handle F60000B5, sequence number 1, ti
mestamp sent 09:59:07 UTC Fri Jan 9 2004, timestamp rcvd 10:12:26 UTC Fri Jan 9
2004
*Jan  9 10:12:26.499: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle F60000B5, sequence number 2, ti
mestamp sent 09:59:07 UTC Fri Jan 9 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
1900
*Jan  9 10:12:26.539: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle F60000B5, sequence number 3, ti
mestamp sent 09:59:08 UTC Fri Jan 9 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
1900
```

Note in the above debug example that the sender handle remains the same and the sequence number increases; also the debugs are taken at the receiver end, and the receiver copies the same handle and sequence number from the echo request to the echo reply.

Sequence Number

The sequence number is assigned by the sender of the MPLS echo request, and can be used to detect missed replies. For example, assume that a ping with a repeat count of 2 is issued. The echo request has a sequence number of 1 and 2. If another ping is issued with the same count, then again the sequence number is 1 and 2, but the handle is different.

The sequence number is assigned by the sender of the MPLS echo request and is copied back in the echo reply. The sequence number is advanced after every echo request. The sequence number to be used is maintained on a per-sender handle basis and not as a global next sequence number.

Timestamp

The timestamp sent is the time inserted by the sender. It is copied back by the receiver in the echo reply. The timestamp received is the time when the echo request is received by the receiver and is put in the echo reply; this field is zero in the echo request. The Timestamp field is in the Network Time Protocol (NTP) time format.

The following debug information taken from the LSPV debug shows the information on the timestamps on the LSP ping:

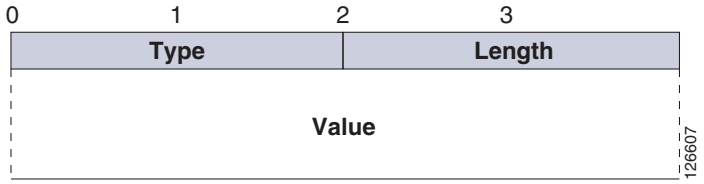
```
Jan  9 10:12:26.495: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2 ,
return_code 0, return_subcode 0, sender handle F60000B5, sequence number 1, timestamp sent
09:59:07 UTC Fri Jan 9 2004, timestamp rcvd 00:00:00 UTC Mon Jan 11900
Jan  9 10:12:26.495: LSPV: Echo Hdr encode: version 1, msg type 2, reply mode 2 ,
return_code 3, return_subcode 0, sender handle F60000B5, sequence number 1, timestamp sent
09:59:07 UTC Fri Jan 9 2004, timestamp rcvd 10:12:26 UTC Fri Jan 9 2004
```

Note that the time is inserted by the sender and is copied back by the receiver.

TLVs

The TLVs field in the LSP echo request/reply packet contains Type-Length-Value (TLV) tuples. TLVs have the format shown in Figure 8.

Figure 8 TLV Format



There are five main TLV types defined for the LSP ping/trace packet, as shown in Table 4. Each of these TLVs are explained in greater detail in the following sections.

Table 4 Five TLV Types

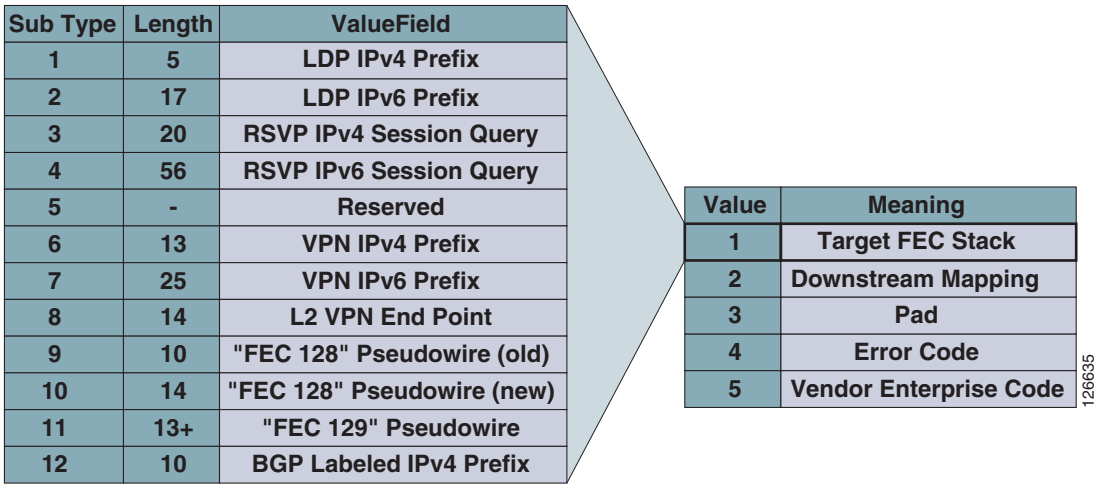
Value	Meaning
1	Target FEC stack
2	Downstream mapping
3	Pad
4	Error code
5	Vendor enterprise code

The value field displays the length in octets. The value field depends on the type; it is zero padded to align to a four-octet boundary.

Target FEC Stack

A target FEC stack is a list of sub-TLVs. The contents of sub-TLVs determine the type of information carried in each sub-TLV. The number of information elements is determined by inspecting the sub-TLV length fields, as shown in Figure 9.

Figure 9 Target FEC Stack

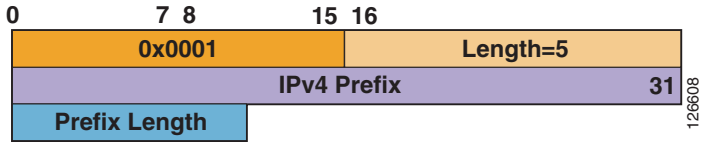


An MPLS echo request contains a target FEC stack that describes the FEC stack being tested. As the name indicates, the target FEC stack TLV defines a stack of FECs/labels. It may contain single or multiple FEC elements or sub-TLVs corresponding to the top or other labels in the packet.

LDP IPv4 Sub-TLV

LDP IPv4 is a type 1 sub-TLV. The value field is 5 bytes long. The value field contains the IPv4 prefix and the prefix length, as shown in Figure 10.

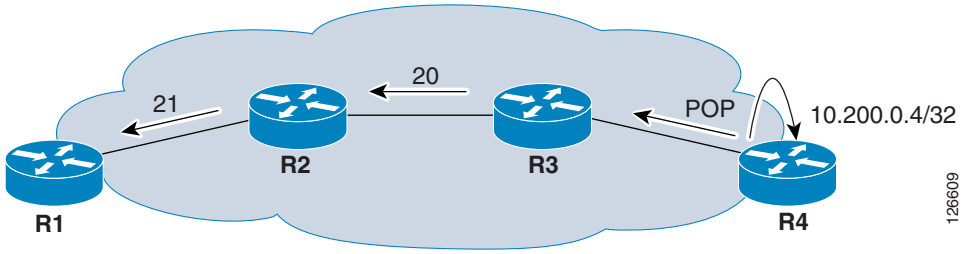
Figure 10 LDP IPv4 Value Field



The IPv4 prefix is in network byte format; that is, if the prefix is shorter than 32, then trailing bits are set to zero. The prefix length is one octet long and indicates the prefix mask in bits (or prefix mask). LDP IPv4 sub-TLV is used to check the liveliness of the LSP established with LDP to reach an IPv4 prefix. The sender of the echo request inserts the IPv4 prefix and mask information corresponding to the LSP being tested.

For example, assume that router R1 has an LDP label binding 21 for prefix 10.200.0.4, as shown in Figure 11.

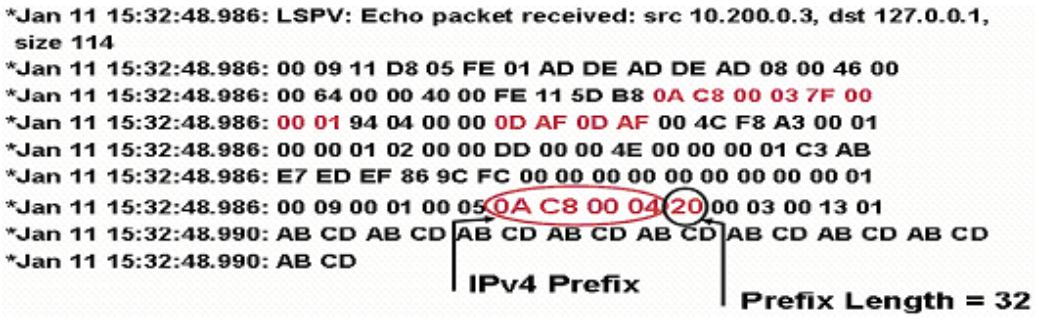
Figure 11 LDP IPv4 Example



To verify that label 21 does indeed reach an egress LSR (R4 in this case), R1 puts the 10.200.0.4/32 prefix address for the selected LSP in ipv4 prefix sub-tlv of the echo request packet.

The example in Figure 12 shows the output of the **debug mpls lspv packet** and **debug mpls lspv tlv** commands for an echo request packet sent from R1 (10.200.0.3) to R4 (10.200.0.4). As indicated in the output, LDP IPv4 contains destination IP address 10.200.0.4 followed by a prefix mask of 32 bits.

Figure 12 Sample Output

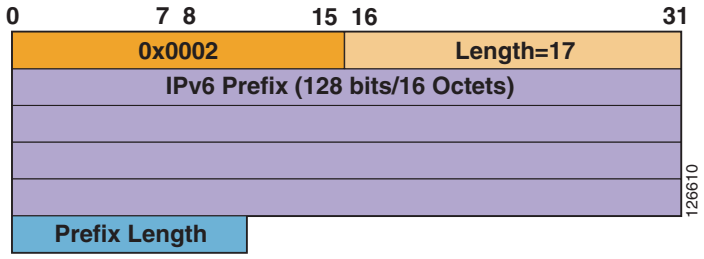



```
*Jan 11 15:32:48.990: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2 ,
return_code 0, return_subcode 0, sender handle DD00004E, sequence number 1, timestamp sent
15:19:09 UTC Sun Jan 11 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1  1900
*Jan 11 15:32:48.990: LSPV: tlvtype 1, tlvlength 9
*Jan 11 15:32:48.990: LSPV: IPV4 FEC decode: destaddr 10.200.0.4/32
*Jan 11 15:32:48.990: LSPV: Target FEC stack length = 9, retcode = 3
*Jan 11 15:32:48.990: LSPV: tlvtype 3, tlvlength 19
*Jan 11 15:32:48.990: LSPV: Pad TLV decode: type 1, size 19
```

LDP IPv6 Sub-TLV

This is a target FEC sub-TLV of type 2. The value consists of 16 octets of an IPv6 prefix followed by one octet of prefix length in bits. The IPv6 prefix is in network byte order; if the prefix is shorter than 128 bits, the trailing bits is set to zero. LDP IPv4 sub-TLV is used to check the liveliness of the LSP established with LDP to reach an IPv6 prefix. The sender of the echo request inserts the IPv6 prefix and mask information corresponding to the LSP being tested. The Cisco IOS software currently does not support LDP IPv6 TLV. LDP IPv6 sub-TLV format is shown in Figure 13.

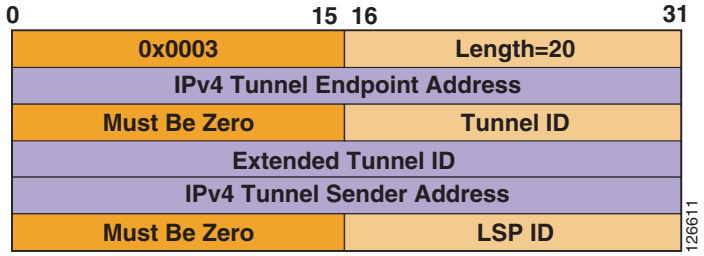
Figure 13 LDP IPv6 sub-TLV Format



RSVP IPv4 Sub-TLV

This is the target FEC stack TLV of type 3. The value field is 20 octets in length and contains five-tuple parameters that uniquely identify a traffic engineered (TE) RSVP tunnel, as shown in Figure 14.

Figure 14 RSVP IPv4 Sub-TLV Value Field



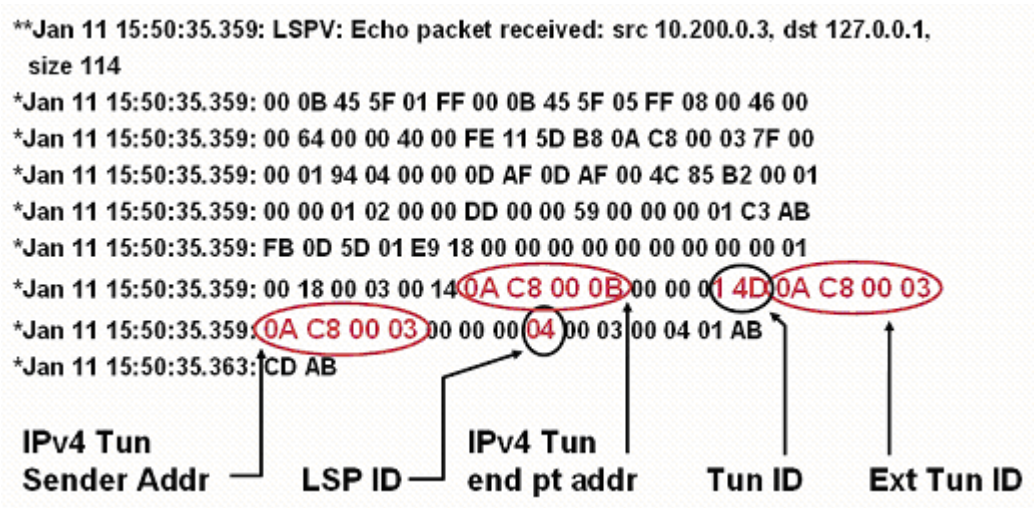
As the name indicates, this TLV is included in the echo packet to verify the tunnel label and check the liveliness of the RSVP tunnel. The five tuples carried in the RSVP IPv4 sub-TLV are defined as follows:

- IPv4 tunnel end point address—IPv4 address of egress node used as the destination address by the TE tunnel.
- Tunnel ID—16-bit identifier used in the SESSION that remains constant over the life of the tunnel. This is usually the tunnel interface number.
- Extended Tunnel ID—32-bit identifier used in the SESSION that remains constant over the life of the tunnel. In the Cisco IOS software, this is normally set to the source IPv4 address of the tunnel interface.
- IPv4 tunnel sender address—IPv4 address for a sender node or the tunnel source address.

- LSP ID—16-bit identifier used in the SENDER_TEMPLATE and the FILTER_SPEC. LSP ID helps in avoiding double-counting the bandwidth, and allows the sharing of bandwidth resources between two tunnels that have all the above mentioned parameters the same except for the LSP ID. LSP ID is incremented when changes happen in the tunnels that cause the tunnel to flap or to re-optimize.

The example in Figure 15 shows the debug outputs of the **mpls lsp packet** and **mpls lsp packet tlv** commands. The output shows an echo packet destined to 10.200.0.11 sourced by 10.200.0.3 to test the RSVP tunnel liveliness.

Figure 15 Debug Outputs




```
*Jan 11 15:50:35.363: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle DD000059, sequence number 1, ti
mestamp sent 16:40:45 UTC Sun Jan 11 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1 1900
*Jan 11 15:50:35.363: LSPV: tlvtype 1, tlvlength 24
*Jan 11 15:50:35.363: LSPV: RSVP IPV4 FEC decode: srcaddr 10.200.0.3, destaddr
10.200.0.11, tun id 333, ext tun id 180879363, lsp id 4
*Jan 11 15:50:35.363: LSPV: Target FEC stack length = 24, retcode = 3
*Jan 11 15:50:35.363: LSPV: tlvtype 3, tlvlength 4
*Jan 11 15:50:35.363: LSPV: Pad TLV decode: type 1, size 4
*Jan 11 15:50:35.363: LSPV: Echo Hdr encode: version 1, msg type 2, reply mode 2
, return_code 3, return_subcode 0, sender handle DD000059, sequence number 1, timestamp
sent 16:40:45 UTC Sun Jan 11 2004, timestamp rcvd 15:50:35 UTC Sun Jan 1
1 2004
```

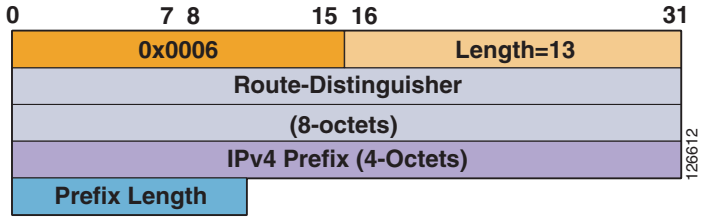
RSVP IPv6 Sub-TLV

This is the type 4 FEC stack sub-TLV, which is used to verify the TE LSP-established IPv6 prefixes. The RSVP IPv6 sub-TLV format is similar to the RSVP IPv4 TLV format, except that the tunnel end address, sender address, and extended tunnel ID are IPv6 addresses. The Cisco IOS software currently does not support this TLV.

VPN IPv4 Prefix Sub-TLV

This is the type 6 for VPN IPv4 prefix TLV. The value field consists of the route distinguisher (RD) advertised with the VPN IPv4 prefix, the IPv4 prefix (with trailing 0 bits to make 32 bits in all), and a prefix length as shown in Figure 16.

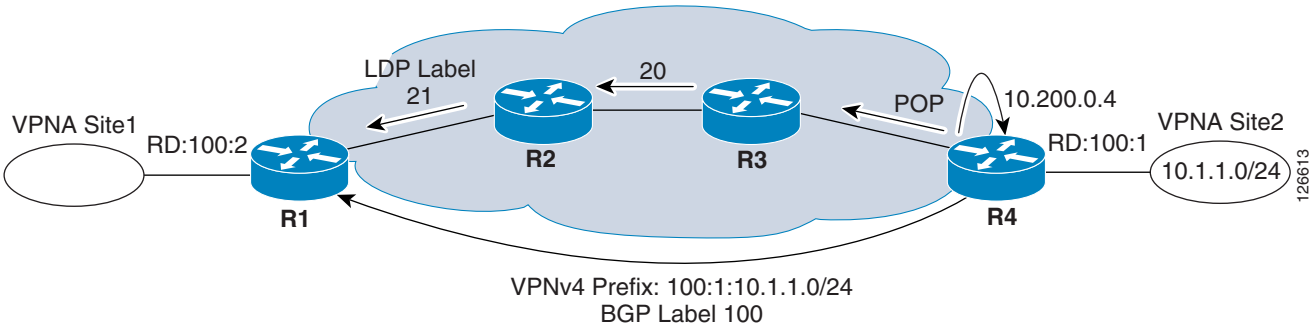
Figure 16 VPN IPv4 Prefix Sub-TLV Value Field



As the name indicates, this TLV is included to verify the VPN label information. For example, assume that R1 has received a label 100 via Multiprotocol Border Gateway Protocol (MP-BGP) for prefix 10.1.1.0/24 in VPNA with RD=100:1. Note that the RD value used by R1 may or may not be the same as the one received from R4 for the same VPN.

To verify that VPN label 100 is the right label to use to reach this VPNv4 prefix, R1 can send an MPLS echo request with an FEC stack TLV containing a single FEC of type VPN IPv4 prefix populated with the VPN prefix 10.1.1.0/24 and an RD of 100:1 information that it received from R4. This is shown in Figure 17.

Figure 17 VPN IPv4 Prefix Sub-TLV Example



Alternatively, R1 can also send an FEC stack TLV with two FECs: the first one of type LDP IPv4 with a prefix of 10.200.0.4/32, and the second one of type IP VPN with a prefix 10.1.1.0/24 and RD=100:1.

The Cisco IOS software does not currently support VPN IPv4 prefix sub-TLV.

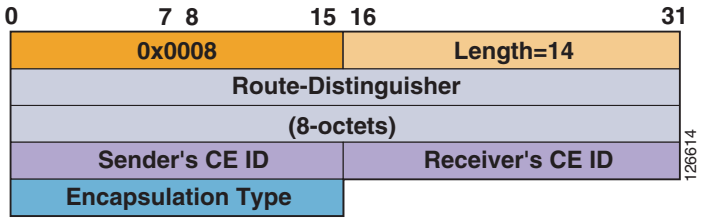
VPN IPv6 Prefix

Similar to the VPN IPv4 prefix, the value field for VPNv6 prefix consists of the RD advertised with the VPN IPv6 prefix, the IPv6 prefix (with trailing 0 bits to make 128 bits in all), and a prefix length. The format is similar to the VPNv4 prefix, and support is not currently available in the Cisco IOS software to test the VPNv6 prefix LSP.

L2 VPN Endpoint

The value field consists of a route distinguisher (8 octets), the sender (of the ping) CE ID (2 octets), the receiver CE ID (2 octets), and an encapsulation type (2 octets), formatted as shown in Figure 18.

Figure 18 L2 VPN Endpoint Value Field

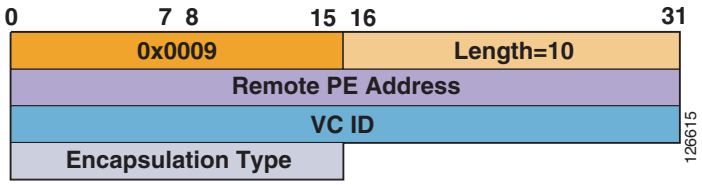


This TLV is not currently supported in the Cisco IOS software.

FEC 128 Pseudowire (L2 Circuit ID Type) Sub-TLV (Old)

The value field consists of the remote PE address (the destination address of the targeted LDP session), a VC ID, and an encapsulation type. The format is shown Figure 19.

Figure 19 FEC 128 Pseudowire (L2 Circuit ID Type) Sub-TLV (Old) Value Field



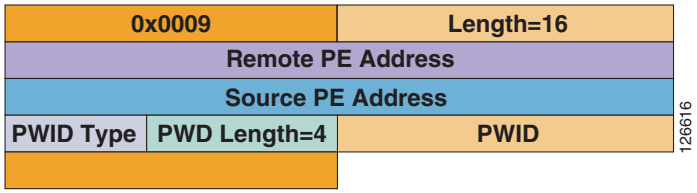
This FEC is no longer used. For backward compatibility, newer LSP ping implementations accept and process this TLV, but are supposed to send LSP ping echo requests with the new TLV, as discussed in the next section (unless explicitly asked by configuration to use the old TLV). Because this TLV does not specify the source address, an LSR receiving this TLV is supposed to use the source IP address of the LSP echo request to infer the PE address of the sender.

Note that by default, the Cisco IOS software currently implements the newer TLV, and that no knob exists to specify the use of the older TLV.

FEC 128 Pseudowire (L2 Circuit ID Type) Sub-TLV (Current)

The purpose of this TLV is to check the liveliness of the L2 circuit FEC or Any Transport over MPLS (AToM) tunnel. The value field consists of the sender PE address (the source address of the targeted LDP session), the remote PE address (the destination address of the targeted LDP session), a PW/VC ID configured for the tunnel, and a PW/VC encapsulation type. The format is shown in Figure 20.

Figure 20 FEC 128 Pseudowire (L2 Circuit ID Type) Sub-TLV (Current) Value Field



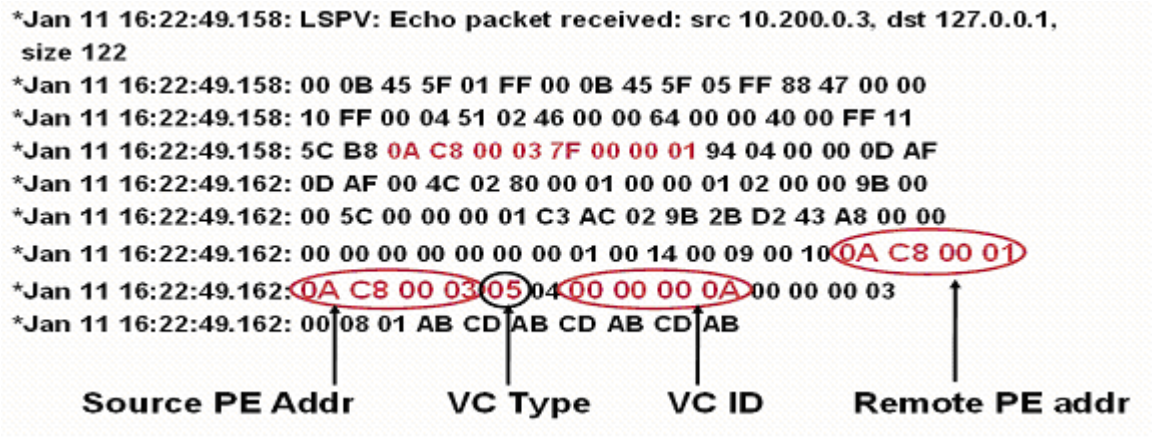
For reference, various currently defined pseudowire (PW) types are listed Figure 21.

Figure 21 PW Types

```
PW type   Description
0x0001    Frame Relay DLCI                          ! Frame Relay
0x0002    ATM AAL5 SDU VCC transport                ! ATM AAL5 SDU
0x0003    ATM transparent cell transport            ! ATM Cell Port Mode
0x0004    Ethernet Tagged Mode                      ! Ethernet VLAN
0x0005    Ethernet                                  ! Ethernet
0x0006    HDLC                                      ! HDLC
0x0007    PPP                                       ! PPP
0x0008    SONET/SDH Circuit Emulation Service Over MPLS (CEM)
0x0009    ATM n-to-one VCC cell transport           ! ATM Cell VC Mode
0x000A    ATM n-to-one VPC cell transport           ! ATM Cell VP Mode
0x000B    IP Layer2 Transport                       ! Interworking IP
0x000C    ATM one-to-one VCC Cell Mode
0x000D    ATM one-to-one VPC Cell Mode
0x000E    ATM AAL5 PDU VCC transport
0x000F    Frame-Relay Port mode
0x0010    SONET/SDH Circuit Emulation over Packet (CEP)
0x0011    Structure-agnostic E1 over Packet (SAToP)
0x0012    Structure-agnostic T1 (DS1) over Packet (SAToP)
0x0013    Structure-agnostic E3 over Packet (SAToP)
0x0014    Structure-agnostic T3 (DS3) over Packet (SAToP)
0x0015    CESoPSN basic mode
0x0016    TDMoIP basic mode
0x0017    CESoPSN TDM with CAS
0x0018    TDMoIP TDM with CAS
```

The example in Figure 22 shows the output of the **debug mpls lspv packet** and **debug mpls lspv tlv** commands.

Figure 22 Debug Output Example



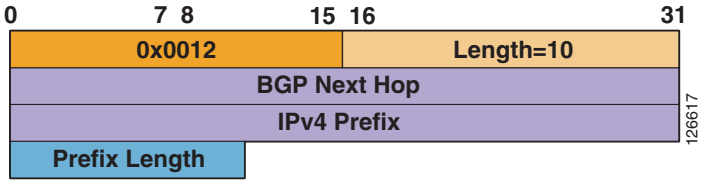

```
*Jan 11 16:22:49.162: LSPV: Echo Hdr decode: version 1, msg type 1, reply mode 2,
return_code 0, return_subcode 0, sender handle 9B00005C, sequence number 1, timestamp sent
17:12:59 UTC Sun Jan 11 2004, timestamp rcvd 00:00:00 UTC Mon Jan 11
*Jan 11 16:22:49.162: LSPV: tlvtype 1, tlvlength 20
*Jan 11 16:22:49.162: LSPV: AToM FEC decode: srcaddr 10.200.0.1, destaddr 10.200.0.3, vcid
10, vctype 5
*Jan 11 16:22:49.162: LSPV: Target FEC stack length = 20, retcode = 3
*Jan 11 16:22:49.162: LSPV: tlvtype 3, tlvlength 8
*Jan 11 16:22:49.166: LSPV: Pad TLV decode: type 1, size 8
*Jan 11 16:22:49.166: LSPV: Echo Hdr encode: version 1, msg type 2, reply mode 2,
return_code 3, return_subcode 0, sender handle 9B00005C, sequence number 1, timestamp sent
17:12:59 UTC Sun Jan 11 2004, timestamp rcvd 16:22:49 UTC Sun Jan 11 2004
```

Note that Pad TLV is used to make the total length a multiple of 4 by adding zeroes. The length field does not include the length of padding.

BGP Labeled IPv4 Prefix

This TLV is used to verify the label FEC advertised using IPv4 Border Gateway Protocol (BGP). The value field consists of the BGP Next Hop associated with the Network Layer Reachability Information (NLRI) advertising the prefix and label, the IPv4 prefix padded with trailing 0 to make 32 bits, and the prefix length. The format is shown in Figure 23.

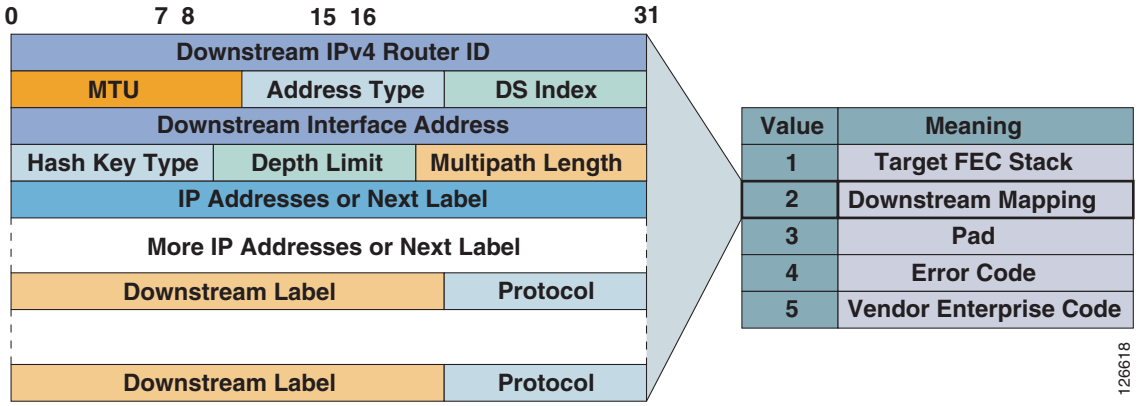
Figure 23 BGP Labeled IPv4 Prefix Value Field



Downstream TLV

The downstream mapping object is an optional TLV. The length is 16 + M + 4*N octets, where M is the multipath length, and N is the number of downstream labels. Figure 24 shows the value field format of a downstream mapping TLV.

Figure 24 Downstream Mapping Value Field



The downstream mapping TLV contains the following fields:

- MTU

  The MTU is the largest MPLS frame (including label stack) that fits on the interface to the downstream LSR. In Cisco terminology, this is referred to as the Maximum Receivable Unit (MRU) and is the biggest packet size that a router can receive and forward downstream without fragmentation.

- Address Type

  The address type indicates whether the interface is numbered or unnumbered, and is set to one of the values shown in Table 5 below.

Table 5 Address Type Values

Type Number	Address Type
1	IPv4
2	Unnumbered
3	IPv6

- Downstream IP Address and Downstream Interface Address

  If the interface to the downstream LSR is numbered, then the address type is set to IPv4 or IPv6. The downstream IP address is set to either the downstream LSR router ID or the interface address of the downstream LSR. The downstream interface address is set to the downstream LSR interface address.

  If the interface to the downstream LSR is unnumbered, the address type is set to be unnumbered, the downstream IP address is set to the downstream LSR router ID (4 octets), and the downstream interface address is set to be the index assigned by the upstream LSR to the interface.

- Multipath Length

  The length in octets of the multipath information.

- Downstream Label(s)

  The set of labels in the label stack as it appears if this router is forwarding the packet through this interface. Any implicit null labels are explicitly included. Labels are treated as numbers; that is, they are right justified in the field.

  A downstream label is 24 bits, in the same format as an MPLS label minus the TTL field; that is, the MS bit of the label is bit 0, the LS bit is bit 19, the EXP bits are bits 20–22, and bit 23 is the S bit.

- Protocol

  The protocol is taken from the table shown in Table 6.

Table 6 Protocol Values

Protocol Number	Signaling Protocol
0	Unknown
1	Static
2	BGP
3	LDP
4	RSVP-TE
5	Reserved

- Depth Limit

  The depth limit is applicable only to a label stack, and is the maximum number of labels considered in the hash; this is set to zero if unspecified or unlimited.

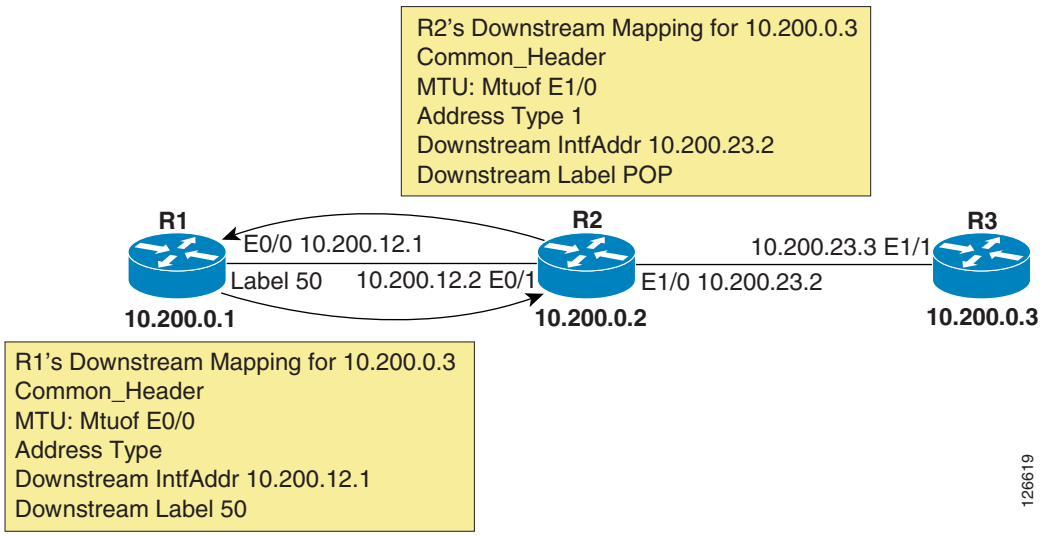
- Multipath Information

The multipath information encodes labels or addresses that exercise this path, and is used for verifying multiple paths (if exit) to get to the destination. The multipath information depends on the hash key type. IP addresses are drawn from the range 127/8. Labels are treated as numbers; that is, they are right justified in the field.

Downstream TLV is included in the echo request in the LSP trace message. The presence of a downstream mapping object is a request to the receiving router to include downstream mapping objects in the echo reply. If the replying router is the destination of the FEC, then a downstream mapping TLV is not included in the echo reply. Otherwise, downstream mapping objects include a downstream mapping object for each interface over which this FEC can be forwarded.

For example, router R1 initiates a trace to destination 10.200.0.3, as shown in Figure 25.

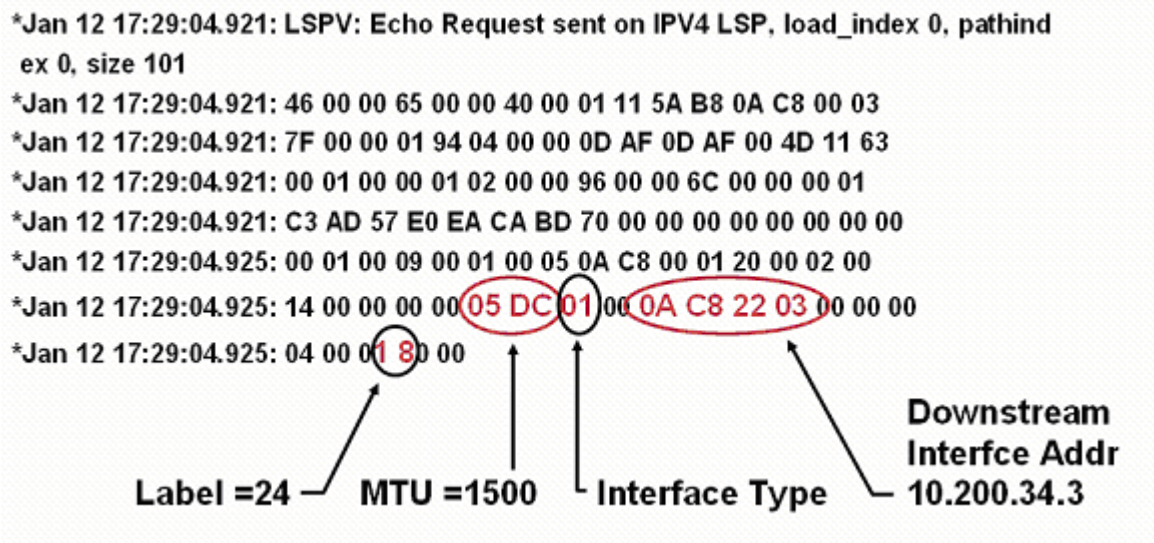
Figure 25 Multipath Information Example



Router 1 includes the downstream TLV objects when sending an echo request downstream towards router R2, including the MTU, address type, downstream interface address, and the corresponding label binding for the destination. Because R2 is not the destination, when the echo request packet is received at R2, it also includes similar information along with the label that it received from R3, and sends the echo reply back to R1.

Figure 26 shows the output of the **debug mpls lspv packet** and **tlv** commands and highlights the contents of the downstream TLV.

Figure 26 Debug Output Example

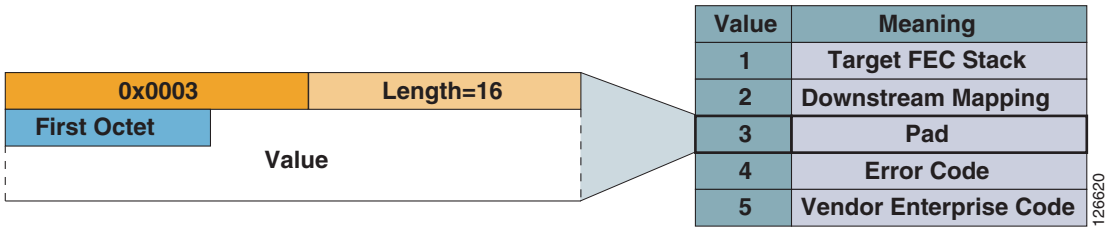



```
*Jan 12 17:29:04.917: LSPV: Echo Hdr encode: version 1, msg type 1, reply mode 2,
return_code 0, return_subcode 0, sender handle 9600006C, sequence number 1, timestamp sent
17:29:04 UTC Mon Jan 12 2004, timestamp rcvd 00:00:00 UTC Mon Jan 11
*Jan 12 17:29:04.917: LSPV: IPV4 FEC encode: destaddr 10.200.0.1/32
*Jan 12 17:29:04.921: LSPV: ds map encode: rtr_id 0, mtu 1500 intf_addr 10.200.34.3 [24]
*Jan 12 17:29:04.921: LSPV: Echo Request sent on IPV4 LSP, load_index 0, pathindex 0, size
101
```

Pad TLV

Pad TLV is used to fill the content of the value field with zeros (or some specified pattern) to make the length a multiple of 4 bytes. The value part of the Pad TLV contains a variable number (>= 1) of octets, as shown in Figure 27.

Figure 27 Pad TLV Value Field



Currently, the first octet takes values of either 1 or 2, specifying whether to drop the Pad TLV from the reply or to copy it in the reply.

Error Code

The Error Code TLV is currently not defined; its purpose is to provide a mechanism for more elaborate error reporting structure if the need arises.

Vendor Enterprise Code

The length for the Vendor Enterprise Code TLV is always 4. The value is the SMI Enterprise code, in network octet order, of the vendor with a vendor private extension to any of the fields in the fixed part of the message, in which case this TLV must be present. This TLV is optional if none of the fields in the fixed part of the message have vendor private extensions.

Generating an LSP Ping

The previous sections of this document have discussed LSP Ping/Traceroute, the information in the echo header, and the information in different TLVs. This section describes how to use LSP Ping.

Operation of LSP Ping

When an LSP ping is issued, an MPLS echo request packet is generated. The payload includes the LDP/RSVP/L2 circuit sub-TLV, depending on the LSP used. Pad TLV is also included in the payload. The type of LSP to ping using the CLI can be chosen as follows:

```
R3#ping mpls ?
  ipv4         Target specified as an IPv4 address
  pseudowire   Target VC specified as an IPv4 address and VC ID
traffic-eng  Target specified as TE tunnel interface
```

The above options choose one of the three TLVs discussed previously. The following sections examine the usage of each of these three options and describe real life case studies.

Using LSP Ping to Verify the Health of an LSP Created by LDP

Assuming that an LSP is created by LDP, use the LDP IPv4 FEC to check the health of this LSP. To do so, choose the **ipv4** keyword option:

ping mpls ipv4 *destination-address destination-mask* [**destination** *address-start address-end increment*] [**ttl** *time-to-live*] [**source** *source-address*] [**repeat** *count*] [**timeout** *seconds*] [{**size** *packet-size*} | {**sweep** *minimum maximum size-increment*}] [**pad** *pattern*] [**reply mode** *reply-mode*] [**interval** *msec*] [**exp** *exp-bits*] [**verbose**]

```
R3#ping mpls ipv4 10.200.0.1/32 ?
  destination  Destination address or address range
  exp          EXP bits in mpls header
  interval     Send interval between requests in msec
  pad          ad TLV pattern
  repeat       Repeat count
  reply        Reply mode
  size         Packet size
  source       Source specified as an IP address
  sweep        Sweep range of sizes
  timeout      Timeout in seconds
  ttl          Time to live
  verbose      Verbose mode for ping output
```

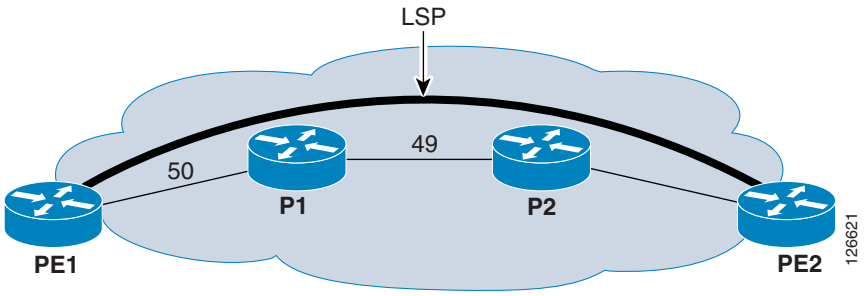
As the above example shows, the Cisco IOS software provides a rich set of options. The values highlighted above are the new ones specific to LSP Ping, and are described as follows:

- Destination address belongs to 127/8 range and the default is 127.0.0.1.
- EXP bits are in a range from 0–7.
- Pad TLV is the pattern that is from 0x0–0xFFFF.
- Size denotes the size of the UDP packet with the label stack imposed. The Pad TLV is used as required to fill the datagram such that the MPLS echo request (UDP packet with label stack) is the specified size.

Case Study 1—MPLS Disabled on the PE

There can be instances in which MPLS has been disabled on a PE because of a misconfiguration or an error condition. For the purposes of this description, see Figure 28.

Figure 28 MPLS Cloud Example

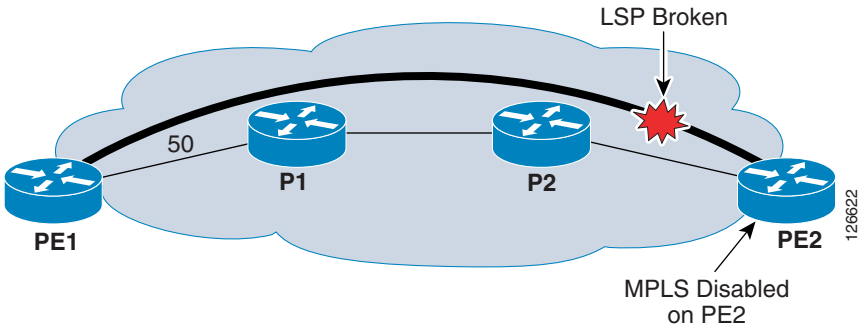


To go from PE1 to PE2, a packet goes through the following process:

1. PE1 pushes a label of 50 and sends the packet to P1.
2. P1 swaps label 50 with label 49.
3. P2 is the PHP router and pops label 49 and sends an IP packet to PE2.

Now assume that MPLS is disabled on PE2, as shown in Figure 29.

Figure 29 MPLS Disabled on PE2



In this case, if a packet is generated for the LSP shown, the following process occurs:

1. PE1 pushes a label of 50 and sends the packet to P1.
2. P1 swaps label 50 with label 49.
3. P2 is the PHP router, but MPLS is disabled on PE2; therefore, P2 receives an implicit null from PE2, so P2 still sends the packet as unlabeled.

If a normal ping is now done, it is successful. If a normal traceroute is done, it too is successful, as shown in the following example:

```
PE1#ping 10.200.0.2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/28/32 ms
PE1#
```

As illustrated in this example, because the LSP is broken at the PHP router and it was supposed to pop off the label either way, you cannot detect the fault using a normal MPLS trace.

If an LSP ping is done from PE1 to PE2, it fails because a certain level of information is checked as part of the normal mechanism of LSP Ping/Traceroute. PE2 receives the echo request, looks at the packet, recognizes that MPLS is not enabled on the routers, and so it replies with a return code of 4, as shown in the following example:

```
PE1#ping mpls ip 10.200.0.2/32
Sending 5, 100-byte MPLS Echos to 10.200.0.1/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
UUUUU
Success rate is 0 percent (0/5)
PE1#
```

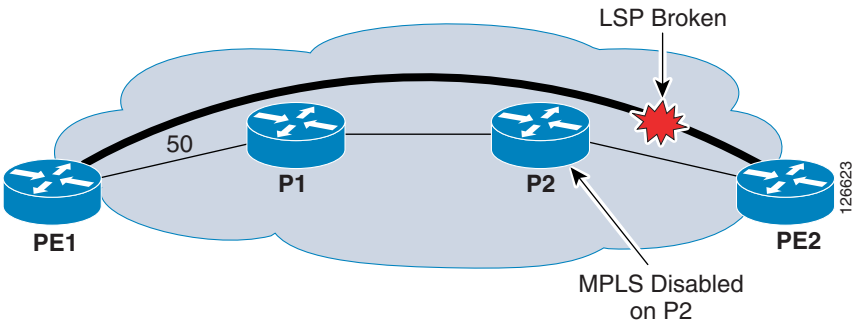
Also, performing an LSP ping with verbose option reveals more information, as in the following example:

```
PE1#ping mpls ipv4 10.200.0.2/32 verbose
Sending 5, 100-byte MPLS Echos to 10.200.0.2/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
U   10.200.22.1, return code 4
U   10.200.22.1, return code 4
U   10.200.22.1, return code 4
U   10.200.22.1, return code 4
U   10.200.22.1, return code 4

Success rate is 0 percent (0/5)
```

Case Study 2—MPLS Disabled on the P Pouter

There are cases where MPLS might be disabled at the P router because of a misconfiguration or an error condition. The same network is assumed as in case study 1, but this time MPLS is disabled on the P router, as shown in Figure 30.

Figure 30 MPLS Disabled on P Router



In this case, a normal ping generated from PE1 to PE2 is successful.

```
PE1#ping 10.200.0.2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/28/32 ms
PE1#
```

Following is the operation on each router:

1. PE1 sends the packet with a label of 50 to P1.
2. P1 in its label forwarding information base (LFIB) has an untagged entry for loopback on PE2. Therefore, P1 pops label 50 off and sends an unlabeled packet to P2.
3. Because P2 is a PHP router, it also sends an unlabeled packet to the destination router PE2, which in turn replies to the ICMP packet.

Now an LSP ping from PE1 to PE2 is unsuccessful:

```
PE1#ping mpls ip 10.200.0.4/32
Sending 5, 100-byte MPLS Echos to 10.200.0.2/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
UUUUU
Success rate is 0 percent (0/5)
PE1#
```

Also, performing an LSP ping with the verbose option reveals more information. Note that in this case, the response comes from P2 and not PE2:

```
PE1#ping mpls ipv4 10.200.0.2/32 verbose
Sending 5, 100-byte MPLS Echos to 10.200.0.2/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target
```

```
Type escape sequence to abort.
U   10.200.12.2, return code 4
U   10.200.12.2, return code 4
U   10.200.12.2, return code 4
U   10.200.12.2, return code 4
U   10.200.12.2, return code 4

Success rate is 0 percent (0/5)
```

Now look at the packet flow. When you issue an LSP ping on PE1, an echo request is generated. The destination address is 127.0.0.1 (default). The actual prefix 10.200.0.2 is part of the payload inside the LDP TLV.

The echo request with destination 127.0.0.1 is sent to P1 with a label 50 pushed on it. On P1, label 50 is associated with an untagged entry because there is no LDP adjacency between P1 and P2.

```
P1#sh mpls for 10.200.0.2
Local  Outgoing    Prefix            Bytes tag  Outgoing   Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
50     Untagged    10.200.0.2/32       0          Se3/0      point2point
P1#
```

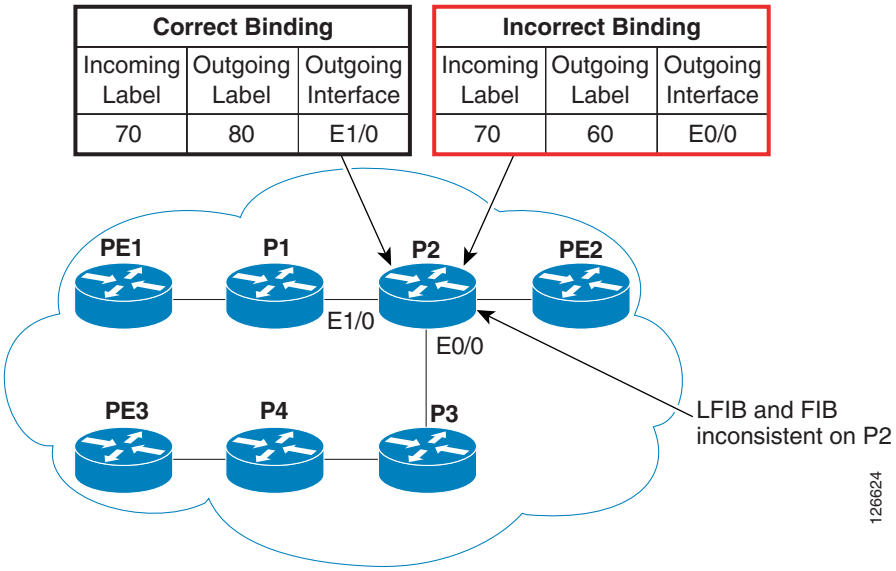
The following operation now occurs:

1. P1 pops off label 50 and sends the echo request to P2 as unlabeled.
2. P2 receives the packet (echo request) and sees that the destination address is 127.0.0.1. P2 punts the packet to the route processor for processing the packet.
3. P2 recognizes that this is an echo request and is destined for the 10.200.0.2 FEC, which does not belong to this router.
4. P2 therefore sends an echo reply back to PE1 with a return code 4.

Case Study 3— Inconsistency Between FIB and LFIB

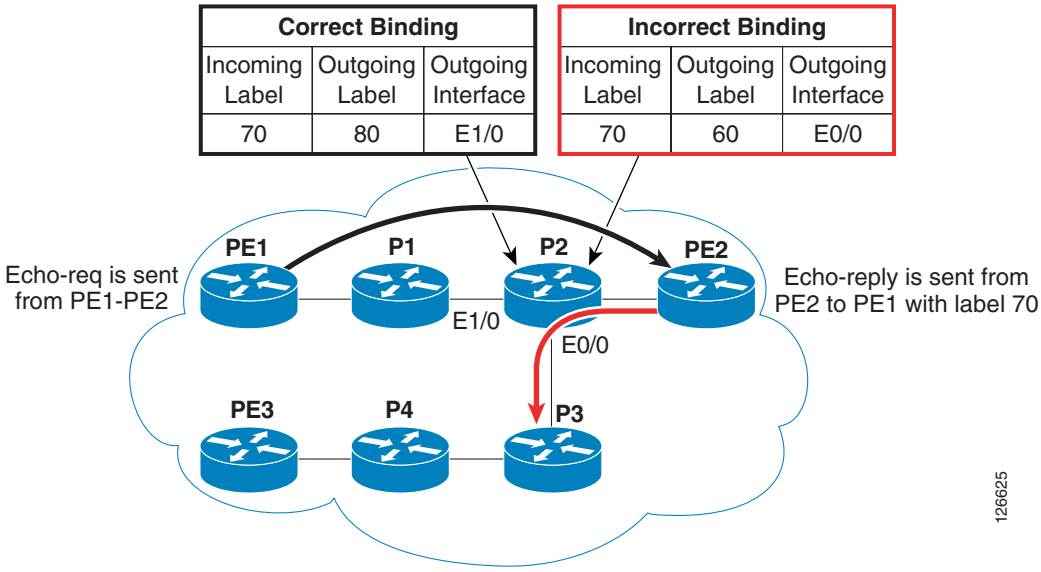
Consider the topology in Figure 31. In this example, there is a customer whose traffic is flowing from PE1 to PE2 using P1 and P2. The customer is complaining that their traffic is dropped somewhere in the SP network. Because of an error condition, an LFIB data corruption occurred on P2. Traffic coming from PE2 to PE1 gets switched to P3 on P2.

Figure 31 Sample Topology for ILFIB Data Corruption



An LSP ping is initiated to troubleshoot the problem, as shown in Figure 32.

Figure 32 Troubleshooting with an LSP Ping



PE2 receives the echo request from PE1 and generates an echo reply back to PE1. The echo reply comes to P2 with top label 70. Because the LFIB is corrupted and has a wrong binding with label 60, it switches the echo reply to P3 with label 60. P3 does not have a label binding for label 60 and drops the packet. This results in the failure of the LSP ping.

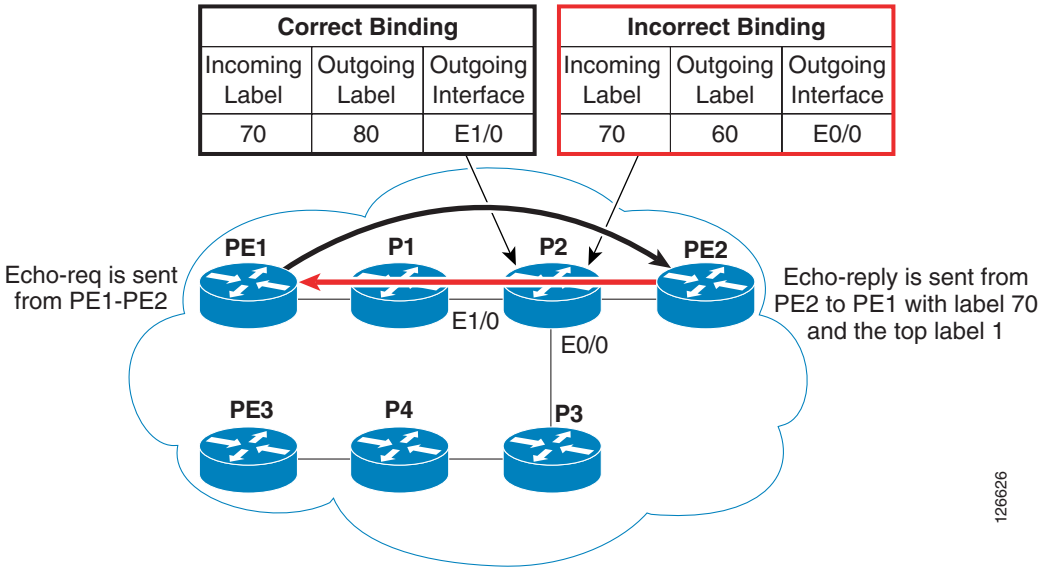
At this point an LSP ping is performed with a router-alert option:

```
PE1#ping mpls ip 10.200.0.1/32 reply mode router-alert
Sending 5, 100-byte MPLS Echos to 10.200.0.3/32,
      timeout is 2 seconds, send interval is 0 msec:
Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/52/60 ms
PE1#
```

Figure 33 shows this sequence. In this case, the router-alert option in the LSP ping is used, so when PE2 generates the echo reply it also pushes an alert label (label 1) on top of label 70. When P2 receives the echo reply, it sees the label 1, which means that this packet is not switched and is processed locally by the router.

The packet is punted to the route processor to be processed. P2 realizes that the packet is an echo reply with the router-alert option and the destination is an IP address on PE1. P2 composes the same echo reply with the same destination address. Before generating the echo reply, P2 looks at the forwarding information base (FIB) entry and sees that it needs to put a label of 80 before sending it to P1.

Figure 33 LSP Ping with Router-Alert Option



The LSP ping with the router-alert option is successful. This indicates that somewhere in the network there is an inconsistency between the FIB and LFIB in the return path. Further troubleshooting to isolate the problem might be needed. The additional steps are covered in some of the later case studies. For example, an LSP ping/trace in the other direction could be generated.

Using LSP Ping to Verify the Health of an LSP Created by RSVP

This section describes how to check the health of a TE tunnel. Remember that a TE tunnel is created by exchange of labels by RSVP. LSP Ping uses RSVP IPv4 sub-TLV for checking the liveliness of a TE tunnel.

The CLI option is as follows:

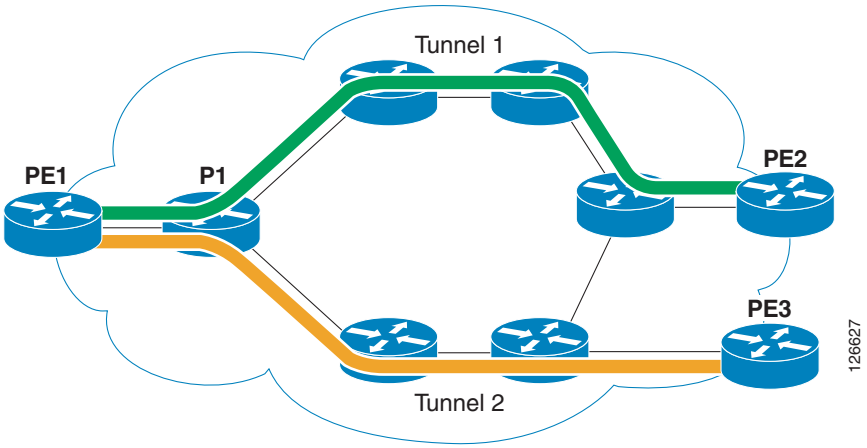
ping mpls traffic-eng Tunnel *tunnel-interface-number* [**ttl** *time-to-live*] [**source** *source-address*] [**repeat** *count*] [**timeout** *seconds*] [{**size** *packet-size*} | {**sweep** *minimum maximum size-increment*}] [**pad** *pattern*] [**reply mode** *reply-mode*] [**interval** *msec*] [**exp** *exp-bits*] [**verbose**]

The details of the above options were explained in the previous section.

Case Study 4—Misrouting into Wrong TE Tunnel (Same Headend with Different Tailend)

Assume that you have two TE tunnels with the same headend but a different tailend, as shown in the topology in Figure 34.

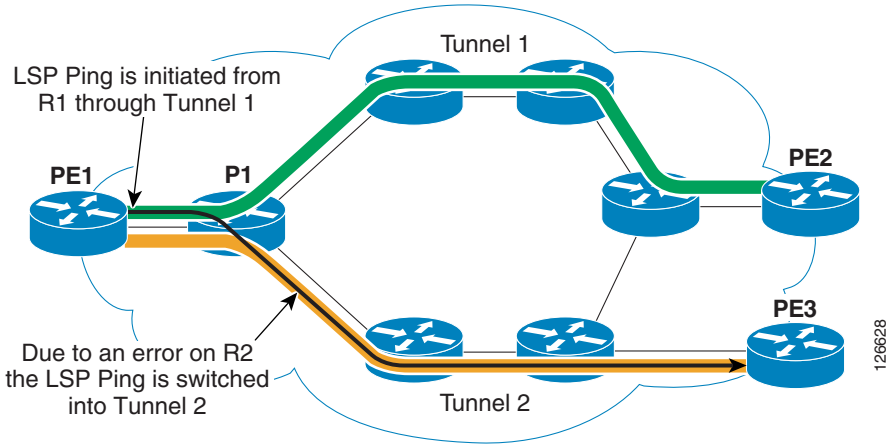
Figure 34 Two TE Tunnels



Customer traffic is coming from PE1 into Tunnel 1 and out of PE2. The customer is complaining of traffic loss. There is an error on P1, and traffic for Tunnel 1 is misrouted into Tunnel 2.

An LSP ping is generated for Tunnel 1 to check its liveliness, as shown in Figure 35.

Figure 35 LSP Ping Check for Tunnel 1



The following sequence of events occurs when issuing an LSP ping:

1. The echo request goes into Tunnel 1 on PE1 and gets to P1.
2. From P1 it gets misrouted into Tunnel 2.
3. The echo request finally gets to PE3. PE3 looks at the five tuples contained in the RSVP TLV. PE3 does not find the five tuples tunnel on the router for which all the five tuples match.
4. PE3 sends back an echo reply to PE1 with a return code of 4, and thus the LSP ping fails.

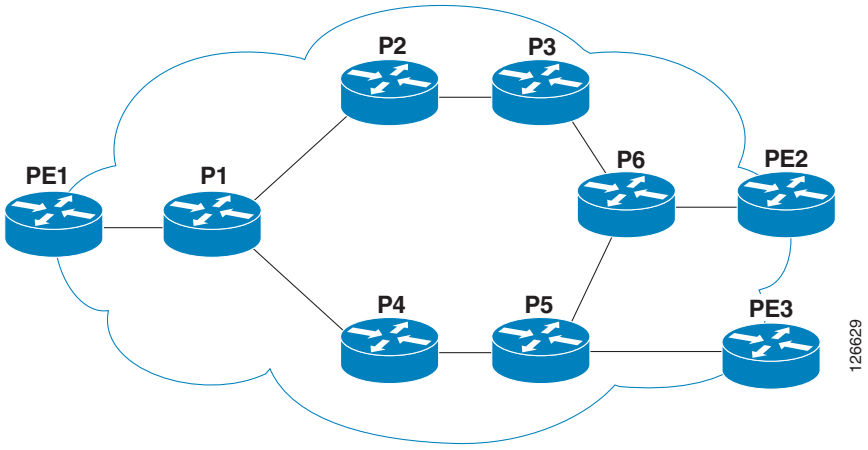
If a verbose option was used for the LSP ping or if relevant debugs were run, you can discover that the echo reply is coming from PE3.

Generating an LSP Trace

The two fundamental types of traces are the following:

- Path trace—Gives information about only one path out of all the possible equal cost multipaths (ECMPs). For example, in Figure 36, a path trace from R1 to R6 gives information about paths R1-R2-R3-R4-R5-R6 *or* R1-R2-R7-R8-R5-R6.
- Tree trace—Returns all possible paths between source and destination. Referring again to Figure 36, when a tree trace is done from R1 to R6, you get information on both the possible paths: R1-R2-R3-R4-R5-R6 *and* R1-R2-R7-R8-R5-R6.

Figure 36 Sample Topology for Two Trace Types



Operation of LSP Traceroute

For LSP Traceroute, MPLS echo requests are generated and the TTL is incremented by 1 on every echo request starting at 1 until the destination is reached. Within the echo request, you have the downstream TLV.

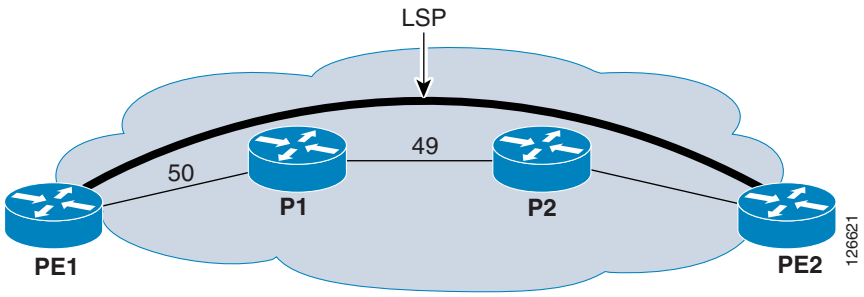
Currently, you can generate an LSP trace for TE tunnels and LSPs created by LDP:

```
R3#traceroute mpls ?
  ipv4         Target specified as an IPv4 address
  traffic-eng  Target specified as TE tunnel interface
```

For the first one, use the LDP sub-TLV, and for the TE tunnel, use the RSVP IPv4 sub-TLV.

To explain the operation of LSP Traceroute, Figure 37 shows an example of a trace generated with LDP sub-TLV.

Figure 37 Sample Topology for LSP Traceroute



Assume that you generate an LSP trace from PE1 to PE2. There is a loopback configured on PE2 with IP address of 10.200.0.2:

```
PE1#tracer mpls ipv4 10.200.0.2/32
Tracing MPLS Label Switched Path to 10.200.0.5/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
  0 10.200.11.1 MRU 4470 [Labels: 50 Exp: 0]
R 1 10.200.12.1 MRU 4470 [Labels: 49 Exp: 0] 8 ms
R 2 10.200.22.2 MRU 4474 [implicit-null] 3 ms
! 3 10.200.22.1 2 ms
PE1#
```

Now look at each step. The **debug mpls lspv tlv** command is enabled on the routers to get more information.

1. PE1 sends an echo request to P1 with LDP sub-TLV and also the downstream TLV. The label on the packet is 50 and the TTL on the label is “1”.

   ```
   PE1#
   *Jun 29 21:40:20.719: LSPV: Echo Hdr encode: version 1, msg type 1, reply mode 2
   , return_code 0, return_subcode 0, sender handle 6800000F, sequence number 1, ti
   mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
    1900
   *Jun 29 21:40:20.719: LSPV: IPV4 FEC encode: destaddr 10.200.0.2/32
   *Jun 29 21:40:20.719: LSPV: ds map encode: rtr_id 0, mtu 4470 intf_addr 122.
   10.200.11.1  [50]
   ```

2. P1 receives the echo request but is able to label-switch the packet because the TTL has expired, so the packet is processed locally.

   P1 recognizes that this is an LSP trace packet for 10.200.0.2. P1 determines the IP address for “outgoing interface” for 10.200.0.2, the MRU of that interface, and the label to be inserted on packets destined for 10.200.0.2. All of this information is put in the downstream TLV and sent back to PE1 in the echo reply.

   Note that the MRU value is put in the MTU field of the downstream TLV, as was previously explained in the Downstream TLV, page 23.

   Now look at the debugs from PE1 when it receives the echo reply:

```
*Jun 29 21:40:20.723: LSPV: Echo Hdr decode: version 1, msg type 2, reply mode 2
, return_code 6, return_subcode 0, sender handle 6800000F, sequence number 1, ti
mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 21:40:18 UTC Tue Jun 2
9 2004
*Jun 29 21:40:20.723: LSPV: tlvtype 2, tlvlength 20
*Jun 29 21:40:20.723: LSPV: Downstream mapping found
*Jun 29 21:40:20.723: LSPV: ds map decode: rtr_id 0, mtu 4470 intf_addr 10.200.12.1
[49]
*Jun 29 21:40:20.723: LSPV: Echo Reply Downstream Mapping TLV
*Jun 29 21:40:20.723: LSPV: remote label stack: 49
```

Note that the return code is 6 in the echo reply that is received by PE1 from P1. Also, the MRU is 4470; this is the MRU of the interface on P1 connected to P2. IP address 10.200.12.1 is of the same interface. Label value 49 is the value of the outgoing label for 10.200.0.2 in the LFIB of P1.

3. PE1 sends another echo request for 10.200.0.2 to P1. The echo request contains LDP sub-TLV and downstream TLV as in Step 1. But the values in the downstream TLV are taken from the one received in Step 2. Also, a label value of 50 is pushed in, but this time with a TTL of 2.

```
*Jun 29 21:40:20.727: LSPV: Echo Hdr encode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle 6800000F, sequence number 2, ti
mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
 1900
*Jun 29 21:40:20.727: LSPV: IPV4 FEC encode: destaddr 10.200.02/32
*Jun 29 21:40:20.727: LSPV: ds map encode: rtr_id 0, mtu 4470 intf_addr 10.200.12.1
[49]
```

4. P1 receives the echo request with a label of 50 and TTL of 2. P1 swaps label 50 with label 49, decrements TTL to 1 and sends the echo request to P2. The TTL would expire on P2 and therefore P2 would have to locally process the packet.

   P2 recognizes that this is an LSP trace packet for 10.200.0.2. P2 determines the IP address for "outgoing interface" for 10.200.0.2, the MRU of that interface, and the label to be inserted on packets destined for 10.200.0.2. All of this information is put in the downstream TLV and sent back to PE1 in the echo reply. Note that the MRU value is put in the MTU field of the downstream TLV, as was previously explained in the Downstream TLV, page 23.

   Now look at the debugs from PE1 when it receives the echo reply:

```
*Jun 29 21:40:20.727: LSPV: Echo Hdr decode: version 1, msg type 2, reply mode 2
, return_code 6, return_subcode 0, sender handle 6800000F, sequence number 2, ti
mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 22:44:27 UTC Tue Jun 2
9 2004
*Jun 29 21:40:20.727: LSPV: tlvtype 2, tlvlength 20
*Jun 29 21:40:20.727: LSPV: Downstream mapping found
*Jun 29 21:40:20.727: LSPV: ds map decode: rtr_id 0, mtu 4474 intf_addr 10.200.22.2
[3]
*Jun 29 21:40:20.727: LSPV: Echo Reply Downstream Mapping TLV
*Jun 29 21:40:20.727: LSPV: remote label stack: 3
```

   As in Step 2, it can be seen that the return code is 6. The MRU is 4474, because P2 is the PHP router. For further explanation of this, please see Downstream TLV, page 23. Also look at the label value. Because P2 is the PHP router, there is a label value of 3, which is the implicit null value.

5. PE1 sends another echo request for 10.200.0.2 to P1. The echo request contains LDP sub-TLV and downstream TLV as in Steps 1 and 3, but the values in the downstream TLV are taken from the one received in Step 4. Also, a label value of 50 is pushed in, but this time with a TTL of 3.

```
*Jun 29 21:40:20.731: LSPV: Echo Hdr encode: version 1, msg type 1, reply mode 2
, return_code 0, return_subcode 0, sender handle 6800000F, sequence number 3, ti
mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 00:00:00 UTC Mon Jan 1
 1900
*Jun 29 21:40:20.731: LSPV: IPV4 FEC encode: destaddr 10.200.0.2/32
```

```
*Jun 29 21:40:20.731: LSPV: ds map encode: rtr_id 0, mtu 4474 intf_addr 10.200.22.2
[3]
```

6. P1 receives the echo request with a label of 50 and TTL of 3. P1 swaps label 50 with label 49, decrements TTL to 2, and sends the echo request to P2. Because P2 is the PHP router, it pops label 49 off and sends the packet to PE2.

   PE2 recognizes that this is an LSP trace packet for 10.200.0.2 and that this is an IP address that belongs to the router. PE2 sends an echo reply back to PE1 with a return code of 3 and without any TLVs.

   Now look at the debugs from PE1 when it receives the echo reply:

```
*Jun 29 21:40:20.731: LSPV: Echo Hdr decode: version 1, msg type 2, reply mode 2
, return_code 3, return_subcode 0, sender handle 6800000F, sequence number 3, ti
mestamp sent 21:40:20 UTC Tue Jun 29 2004, timestamp rcvd 06:21:44 UTC Sun Jun 2
4 2001
```

Using LSP Traceroute to Verify the Health of an LSP Created by LDP

Assume that you have an LSP created by LDP. To check the health of this LSP, use the LDP IPv4 FEC by choosing the **ipv4** keyword option:

trace mpls {**ipv4** *destination-address destination-mask* [**destination** *address-start address-end address-increment*] [**source** source-*address*]
[**timeout** *seconds*] [**reply mode** *reply-mode*] [**ttl** *maximum-time-to-live*] [**exp** *exp-bits*]

```
PE1#traceroute mpls ipv4 10.200.0.3/32 ?
  destination  Destination address or address range
  exp               EXP bits in mpls header
  reply            Reply mode
  source         Source specified as an IP address
  timeout       Timeout in seconds
  ttl                Maximum time to live
```

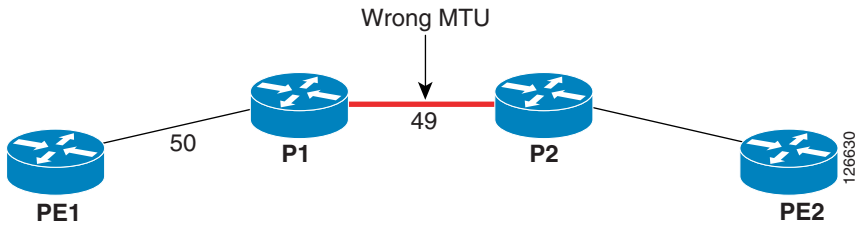
Note that, as with LSP Ping, the Cisco IOS software provides a rich set of options for LSP Traceroute. Some of the new options specific to LSP Ping are highlighted above and are explained as follows:

- Destination address is from 127/8 range and the default is 127.0.0.1.
- EXP bits are in a range from 0–7

Case Study 5—Wrong MTU between Routers

The customer is using an LSP PE1-P1-P2-PE2 (see Figure 38), and is complaining about intermittent response for the traffic.

Figure 38 Example of Wrong MTU between Routers



A sweeping LSP ping reveals that packets over 1500 bytes are failing. You detect a problem but are not able to isolate the problem.

A normal traceroute gives the following output:

```
PE1#traceroute 10.200.0.2

Type escape sequence to abort.
Tracing the route to 10.200.0.2

  1 10.200.11.1 [MPLS: Label 50 Exp 0] 0 msec 0 msec 0 msec
  2 10.200.12.1 [MPLS: Label 49 Exp 0] 0 msec 0 msec 0 msec
  3 10.200.22.1 0 msec *  0 msec
PE1#
```

You still do not know exactly where the problem is. An LSP trace reveals the following information:

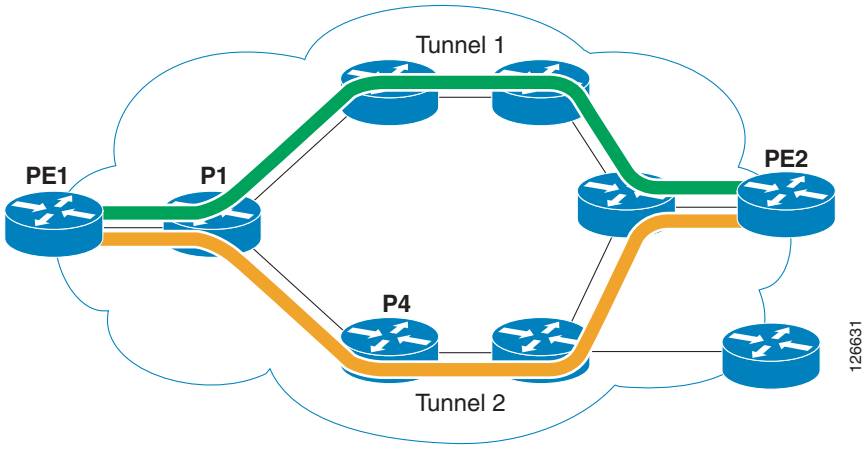
```
PE1#tracer mpls ipv4 10.200.0.2/32
Tracing MPLS Label Switched Path to 10.200.0.5/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
  0 10.200.11.1 MRU 4470 [Labels: 50 Exp: 0]
R 1 10.200.12.1 MRU 1500 [Labels: 49 Exp: 0] 8 ms
R 2 10.200.22.2 MRU 4474 [implicit-null] 3 ms
! 3 10.200.22.1 2 ms
PE1#
```

This shows that the issue is easily isolated using LSP Traceroute to be on P1–P2 link.

Case Study 6—Misrouting into Wrong TE Tunnel (Same Headend and Tailend)

This case study examines a topology with two TE tunnels with the same headend and tailend, as shown in Figure 39.

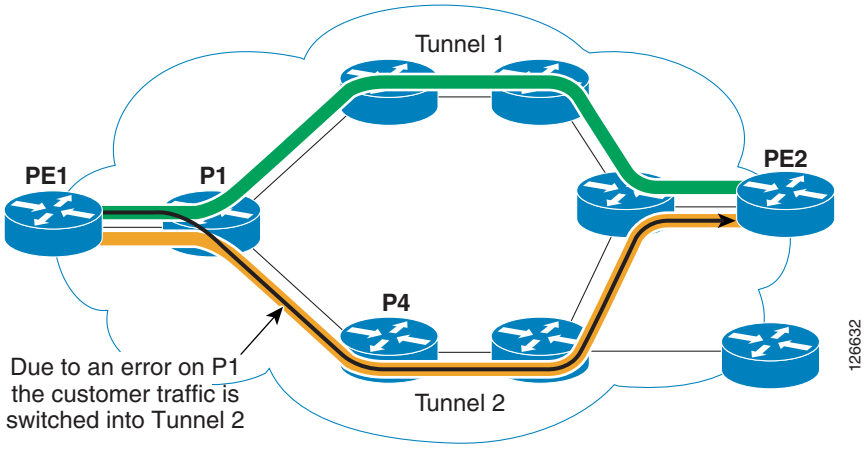
Figure 39 Two TE Tunnels with Same Headend and Tailend



As in case study 4, customer traffic is coming from PE1 into Tunnel 1 and out of PE2. The customer is complaining of not getting the desired service as specified in the service level agreement (SLA). There is an error on P1 and traffic for Tunnel 1 is misrouted into Tunnel 2.

An LSP ping is generated for Tunnel 1 to check its liveliness, as shown in Figure 40.

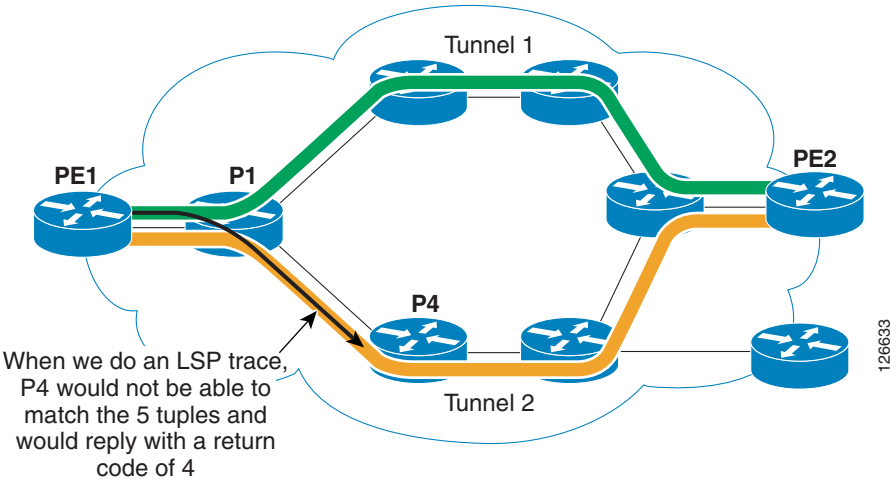
Figure 40 LSP Ping on Tunnel 1



As shown in Figure 40, the LSP ping packet gets misrouted into Tunnel 2 on P1, but because the tailend for Tunnel 2 is also PE2, the LSP ping lands on PE2. Now PE2 looks at the RSVP sub-TLV of the echo request and sees that all the five tuples match with Tunnel 1 on PE2. Therefore, PE2 sends an echo reply with a return code of 3. The operator is not able to determine that the LSP ping was misrouted into Tunnel 2.

The next thing to do is to generate an LSP trace, as shown in Figure 41.

Figure 41 LSP Trace



As shown in Figure 41, when an LSP trace is generated, an echo request is sent to P1 with a TTL value of 1 for the outermost label. P1 replies back accordingly, as explained in Understanding Traditional Ping and Trace, page 3. The following sequence then takes place:

1. PE1 increments the TTL and sends the packet to P1, which in turn swaps the outermost label so that the packet is switched into Tunnel 2 and the echo request reaches P4.
2. P4 then receives the packet with TTL 1, and therefore punts the packet for local processing.

3. P4 looks at the five tuples contained in the RSVP sub-TLV. Because P4 is not a headend/tailend/midpoint for Tunnel 1, it does not find any matching TE tunnel on the router for the five tuples in the echo request.
4. P4 therefore replies back to PE1 with a return code of 4, and an operator can now figure out where the problem is.

Using Equal Cost Multipaths

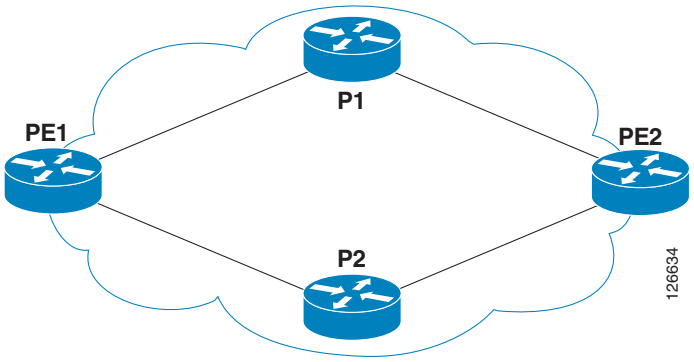
This section first examines how an MPLS packet is handled in an equal-cost multipath (ECMP) environment, and then describes how LSP Traceroute can be used in this environment. As described in Generating an LSP Trace, page 36, there are two types of traces: path and tree trace. The initial release of LSP Traceroute only supports path trace.

Assume that you receive an MPLS packet, and that there are multiple paths that the packet can take while leaving the router. The router then looks for 0x4 under the bottom of the label stack. The first four bits in the IP header indicate the version. 0x4 means that this is an IPv4 packet. If it is IP, then the router performs load balancing based on the IP source and destination address (this is hardware-dependent).

If it is not IP, then the router performs load balancing depending on the bottommost label. For example, in the case of AToM, the control word is different from 0x4 and the bottommost label is the VC label. Remember that some platforms cannot go all the way down the label stack, so the bottommost label might not be a VC label.

Now consider the following example, where PE1 has two paths to reach PE2, as shown in Figure 42.

Figure 42 Equal Cost Multipath Example



Looking at the LFIB, you can see two paths:

```
PE1#sh mpls forwarding-table 10.200.0.1
Local  Outgoing    Prefix           Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched   interface
27     20          10.200.0.1/32          0              PO0/0      point2point
         23          10.200.0.1/32          0              PO1/0      point2point
PE1#
```

If you now generate an LSP trace and use the default destination address of 127.0.0.1, you see that you take the path through P1:

```
PE1# traceroute mpls ip 10.200.0.1/32  destination 127.0.0.1
Tracing MPLS Label Switched Path to 10.200.0.1/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
```

```
       'R' - downstream router but not target

Type escape sequence to abort.
  0 10.200.11.1 MRU 4470 [Labels: 20 Exp: 0]
R 1 10.200.22.2 MRU 4474 [implicit-null] 3 ms
!  2 10.200.22.1 2 ms
PE1#
```

If you now change the destination address to 127.0.0.2, you can see that the path is still through P1:

```
PE1# traceroute mpls ip 10.200.0.1/32  destination 127.0.0.2
Tracing MPLS Label Switched Path to 10.200.0.2/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
  0 10.200.11.1 MRU 4470 [Labels: 20 Exp: 0]
R 1 10.200.22.2 MRU 4474 [implicit-null] 3 ms
!  2 10.200.22.1 2 ms
PE1#
```

The reason to choose the same path with destination address 127.0.0.2 is because the hash between the source and the destination address results in the same output interface. You can now try 127.0.0.3 and find out that you now take the path through P2:

```
PE1# traceroute mpls ip 10.200.0.1/32  destination 127.0.0.3
Tracing MPLS Label Switched Path to 10.200.0.3/32, timeout is 2 seconds

Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target

Type escape sequence to abort.
  0 10.200.12.1 MRU 4470 [Labels: 23 Exp: 0]
R 1 10.200.122.2 MRU 4474 [implicit-null] 3 ms
!  2 10.200.122.1 2 ms
PE1#
```

To check all the possible paths that an LSP takes, change the destination address.

References

[1] draft-ietf-mpls-icmp-02.txt at the following website:
http://www.ietf.org/proceedings/00dec/I-D/draft-ietf-mpls-icmp-02.txt

[2] draft-ietf-mpls-label-encaps-08.txt at the following website:
http://www.potaroo.net/ietf/all-ids/draft-ietf-mpls-label-encaps-08.txt-47028.txt

[3] draft-ietf-mpls-lsp-ping-06.txt at the following website:
http://www.ietf.org/proceedings/04aug/I-D/draft-ietf-mpls-lsp-ping-06.txt