

# Implementing an MPLS VPN over TE Tunnels

Document ID: 29828

---

## Introduction

### Prerequisites

- Requirements
- Components Used
- Conventions
- Background Theory

### Initial VPN Setup Between CE1 and CE2 Without a TE Tunnel

- Topology
- Configuration
- Verification

### Case 1: VPN over a TE Tunnel When the TE Tunnel Is from PE1 to PE2

- Topology
- Configuration
- Verification

### Case 2: VPN over a TE Tunnel When the TE Tunnel Is from PE1 to P2

- Topology
- Configuration
- Verification
- Explanation
- Solution

### Case 3: VPN Between CE1 and CE2 over a TE Tunnel from P1 to P2 When TDP/LDP

#### Is Not Enabled

- Topology
- Configuration
- Verification
- Solution

### Case 4: VPN over a TE Tunnel Between P1 and P2 with LDP Enabled

- Topology
- Configuration
- Verification

### Case 5: MPLS VPN over a Tunnel Between P1 and PE2

- Topology
- Configuration
- Verification

### Known Issues

### Conclusion

### Related Information

---

## Introduction

This document provides sample configurations for implementing a Multiprotocol Label Switching (MPLS) VPN over traffic engineering (TE) tunnels in an MPLS network. In order to gain the benefits of an MPLS VPN over TE tunnels, both should coexist in the network. This document illustrates various scenarios that explain why packet forwarding within an MPLS VPN over TE tunnels might fail. It also provides a possible solution.

# Prerequisites

## Requirements

Readers of this document should have knowledge of these topics:

- MPLS Traffic Engineering and Enhancements
- Configuring a Basic MPLS VPN

## Components Used

This document is not restricted to specific software and hardware versions.

## Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

## Background Theory



As shown in this topology, in a simple MPLS VPN configuration, Provider Edge 1 (PE1) learns the VPN label (Label 1 [L1]) for the VPN prefix 172.16.13.0/24 via multiprotocol Border Gateway Protocol (MPBGP) from PE2 directly, with the next hop as the PE2 loopback address. PE1 also learns the label (L2) for the PE2 loopback address via Label Distribution Protocol (LDP) from its next hop P1.

When forwarding data to the VPN prefix 172.16.13.13, PE1 uses a label stack {L2 L1} with L2 as the outer label. L2 gets swapped by the transit label switch router (LSR), P1. P2 pops the outer L2 and forwards the packet to PE2 with only one L1. To better understand why P2 pops L2, refer to section 3.16 about penultimate hop popping (PHP) in RFC 3031 . Thus, packets to the VPN IP version 4 (IPv4) prefix 172.16.13.0/24 are label switched over an MPLS network.

The MPLS VPN forwarding operation fails if any P router receives the packet with L1 (VPN label) as the only outer label instead of the {L2 L1} label stack. This occurs because none of the P routers has L1 in its label forwarding information base (LFIB) to switch the packet.

An MPLS TE uses Resource Reservation Protocol (RSVP) to exchange labels. When a router is configured for both TE and Tag Distribution Protocol (TDP)/LDP, the router receives different labels from both LDP and RSVP for a given prefix. The labels from LDP and RSVP do not need to be the same in all situations. The router installs an LDP label in the forwarding table if the prefix is learned through an LDP interface, and it installs the RSVP label in the forwarding table if the prefix is learned over a TE tunnel interface.

In the case of a plain TE tunnel (without LDP/TDP enabled on the tunnel), the ingress LSR (the LSR on the headend of the TE tunnel) uses the same label as is used for reaching the tailend of the TE tunnel for all the routes that are learned through a TE tunnel.

For example, there is a TE tunnel from PE1 to P2 learning the prefix 10.11.11.11/32 over the tunnel. The tunnel tailend on P2 is 10.5.5.5, and the label to reach 10.5.5.5 in PE1 is L3. PE1 then uses L3 to reach the destination 10.11.11.11/32, learned over the TE tunnel.

In the scenario above, when there is a TE tunnel between PE1 and P2, consider that PE1 forwards data to Customer Edge 2 (CE2). If L4 is the VPN label, PE1 forwards the data with the label stack {L3 L4}. P1 pops L3, and P2 receives the packet with L4. PE2 is the only LSR that can correctly forward the packet with the outer label L4. P2 does not have an MPBGP session with PE2, so it does not receive the L4 from PE2. Therefore, P2 does not have any knowledge of L2, and it drops the packet.

The configurations and **show** outputs that follow demonstrate this and illustrate one possible solution to this problem.

## Initial VPN Setup Between CE1 and CE2 Without a TE Tunnel

### Topology



### Configuration

Only the relevant parts of the configuration files are included here:

PE1
<pre> hostname PE1 ip cef ! ip vrf aqua   rd 100:1   route-target export 1:1   route-target import 1:1 ! mpls traffic-eng tunnels ! interface Loopback0   ip address 10.2.2.2 255.255.255.255   no ip directed-broadcast ! interface Ethernet2/0/1   ip vrf forwarding aqua   ip address 172.16.1.2 255.255.255.0 ! interface Ethernet2/0/2   ip address 10.7.7.2 255.255.255.0   ip router isis   mpls traffic-eng tunnels   tag-switching ip ! router isis   passive-interface Loopback0   net 47.1234.2222.2222.2222.00   is-type level-1   metric-style wide   mpls traffic-eng router-id Loopback0   mpls traffic-eng level-1 ! router bgp 1   bgp log-neighbor-changes </pre>

```

neighbor 10.11.11.11 remote-as 1
neighbor 10.11.11.11 update-source Loopback0
!
address-family vpnv4
neighbor 10.11.11.11 activate
neighbor 10.11.11.11 send-community extended
exit-address-family
!
address-family ipv4
neighbor 10.11.11.11 activate
no auto-summary
no synchronization
exit-address-family
!
address-family ipv4 vrf aqua
redistribute connected
no auto-summary
no synchronization
exit-address-family

```

## PE2

```

hostname PE2
!
ip vrf aqua
rd 100:1
route-target export 1:1
route-target import 1:1
!
mpls traffic-eng tunnels
!
interface Loopback0
ip address 10.11.11.11 255.255.255.255
!
interface POS0/1
ip address 10.12.12.10 255.255.255.0
ip router isis
mpls traffic-eng tunnels
tag-switching ip
crc 16
clock source internal
!
interface POS5/1
ip vrf forwarding aqua
ip address 172.16.13.11 255.255.255.0
crc 32
clock source internal
!
router isis
passive-interface Loopback0
mpls traffic-eng router-id Loopback0
mpls traffic-eng level-1
net 47.1234.1010.1010.1010.00
is-type level-1
metric-style wide
!
router bgp 1
bgp log-neighbor-changes
neighbor 10.2.2.2 remote-as 1
neighbor 10.2.2.2 update-source Loopback0
no auto-summary
!
address-family vpnv4
neighbor 10.2.2.2 activate
neighbor 10.2.2.2 send-community extended

```

```
exit-address-family
!
address-family ipv4 vrf aqua
redistribute connected
no auto-summary
no synchronization
exit-address-family
!
```

## Verification

PE2 learns PE1 VPN IPv4 prefix 172.16.1.0/24 over MPBGP peering between PE1 and PE2. This is shown here:

```
PE2# show ip route vrf aqua
```

```
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, 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
```

```
10.0.0.0/24 is subnetted, 2 subnets
```

```
B    172.16.1.0 [200/0] via 10.2.2.2, 16:09:10
C    172.16.13.0 is directly connected, POS5/1
```

Similarly, PE1 learns PE2 VPN IPv4 prefix 172.16.13.0/24 over MPBGP peering between PE1 and PE2. This is shown here:

```
PE1# show ip route vrf aqua
```

```
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, 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
```

```
10.0.0.0/24 is subnetted, 2 subnets
```

```
B    172.16.13.0 [200/0] via 10.11.11.11, 16:09:49
C    172.16.1.0 is directly connected, Ethernet2/0/1
```

```
PE1# show ip route vrf aqua 172.16.13.13
```

```
Routing entry for 172.16.13.0/24
```

```
Known via "bgp 1", distance 200, metric 0, type internal
```

```
Last update from 10.11.11.11 16:13:19 ago
```

```
Routing Descriptor Blocks:
```

```
* 10.11.11.11 (Default-IP-Routing-Table), from 10.11.11.11, 16:13:19 ago
```

```
Route metric is 0, traffic share count is 1
```

```
AS Hops 0, BGP network version 0
```

```
PE1# show ip cef vrf aqua 172.16.13.13
```

```
172.16.13.0/24, version 11, cached adjacency 10.7.7.7
```

```
0 packets, 0 bytes
```

```
tag information set
```

```
local tag: VPN route head
```

```
fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}
```

```
via 10.11.11.11, 0 dependencies, recursive
```

```
next hop 10.7.7.7, Ethernet2/0/2 via 10.11.11.11/32
```

```
valid cached adjacency
```

```

tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}

!--- The label stack used to reach 172.16.13.13 is
!--- {17 12308}, where 17 is the outer label to reach next hop 10.11.11.11
!--- and 12308 is the VPN IPv4 label for 172.16.13.0/24.

PE1# show ip cef 10.11.11.11
10.11.11.11/32, version 31, cached adjacency 10.7.7.7
0 packets, 0 bytes
tag information set
  local tag: 21
  fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17}
  via 10.7.7.7, Ethernet2/0/2, 1 dependency
  next hop 10.7.7.7, Ethernet2/0/2
  valid cached adjacency
  tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17}

!--- Outer label 17 is used to reach next hop 10.11.11.11.

```

Thus, CE1 can reach 172.16.13.13 on the CE2 network via the VPN routing and forwarding (VRF) instance "aqua", which is configured on PE1 using the label stack {17 12308}, as shown above.

This **ping** output confirms the connectivity:

```

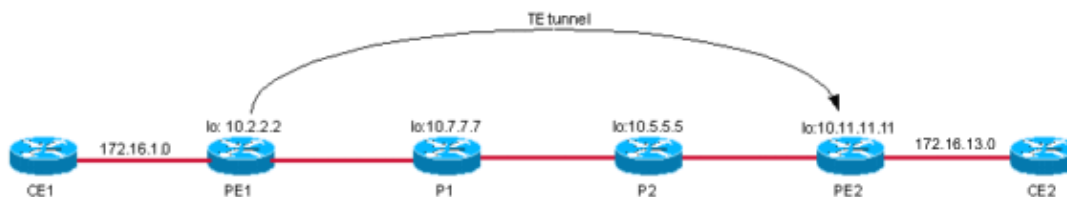
CE1# ping 172.16.13.13

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

```

## Case 1: VPN over a TE Tunnel When the TE Tunnel Is from PE1 to PE2

### Topology



When the TE tunnel is built between the PE routers with autoroute announce used, the egress PE BGP next hop is reachable via the TE tunnel interface. Thus, PE1 uses the TE label to reach PE2.

**Note:** MPLS TE is independent of LDP, which means that, if you have a full mesh of tunnels from PE to PE, you can effectively disable LDP in the routers and do not need to run LDP on the TE tunnel interfaces. However, you must build all tunnels to the BGP next hop of the VPN version 4 (VPNv4) routes. In the example in this Configuration, you can see that this BGP next hop is the Loopback0 on PE2, 10.11.11.11. This same loopback is also the tunnel destination for the tunnel from PE1 to PE2. This explains why, in this example, if there is also a tunnel from PE2 to PE1 for the return traffic, you can disable LDP in the core. Then, forwarding from CE to CE works with all VPNv4 traffic carried over the TE tunnels. If the BGP next hop is not the same as the TE tunnel destination, LDP must be run in the core and on the TE tunnel.

## Configuration

The additional configuration on PE1 to establish a PE tunnel is shown here:

```
PE1
PE1# show run interface tunnel 0
!
interface Tunnel0
 ip unnumbered Loopback0
 no ip directed-broadcast
 no ip route-cache distributed
 tunnel destination 10.11.11.11
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng path-option 10 dynamic
end
```

## Verification

```
PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 11
0 packets, 0 bytes
 tag information set
   local tag: VPN route head
   fast tag rewrite with Tu0, point2point, tags imposed {19 12308}
 via 10.11.11.11, 0 dependencies, recursive
   next hop 10.11.11.11, Tunnel0 via 10.11.11.11/32
   valid adjacency
   tag rewrite with Tu0, point2point, tags imposed {19 12308}
```

```
!--- The label stack to reach 172.16.13.13 is {19 12308}.
!--- BGP next hop for the VPNv4 prefix is 10.11.11.11, which is
!--- the same as the TE tunnel destination.
```

```
PE1# show ip route 10.11.11.11
Routing entry for 10.11.11.11/32
  Known via "isis", distance 115, metric 40, type level-1
  Redistributing via isis
  Last update from 10.11.11.11 on Tunnel0, 00:02:09 ago
  Routing Descriptor Blocks:
    * 10.11.11.11, from 10.11.11.11, via Tunnel0
```

```
!--- The route is via Tunnel0.
```

```
Route metric is 40, traffic share count is 1
```

Now, confirm the outer label used to reach the next hop 10.11.11.11 via Tunnel0.

```
PE1# show mpls traffic-eng tunnels tunnel 0

Name: PE1_t0 (Tunnel0) Destination: 10.11.11.11
Status:
  Admin: up Oper: up Path: valid Signalling: connected

  path option 10, type dynamic (Basis for Setup, path weight 30)

Config Parameters:
  Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 0 bw-based
  auto-bw: disabled
```

```
InLabel : -
OutLabel : Ethernet2/0/2, 19
```

```
!--- Label 19 from RSVP is used to reach destination 10.11.11.11/32.
```

```
RSVP Signalling Info:
  Src 10.2.2.2, Dst 10.11.11.11, Tun_Id 0, Tun_Instance 31
RSVP Path Info:
  My Address: 10.7.7.2
  Explicit Route: 10.7.7.7 10.8.8.7 10.8.8.5 10.12.12.10
                  10.11.11.11
  Record Route: NONE
  Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
RSVP Resv Info:
  Record Route: NONE
  Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=Inf
Shortest Unconstrained Path Info:
  Path Weight: 30 (TE)
  Explicit Route: 10.7.7.2 10.7.7.7 10.8.8.7 10.8.8.5
                  10.12.12.10 10.11.11.11
History:
  Tunnel:
    Time since created: 17 hours, 17 minutes
    Time since path change: 32 minutes, 54 seconds
  Current LSP:
    Uptime: 32 minutes, 54 seconds
  Prior LSP:
    ID: path option 10 [14]
    Removal Trigger: tunnel shutdown
```

Another way to view this information quickly is to use the output modifiers in the **show** commands, as shown here:

```
PE1# show mpls traffic-eng tunnels tunnel 0 | include Label
InLabel : -
OutLabel : Ethernet2/0/2, 19
```

```
!--- This is the label to reach 10.11.11.11.
```

Look at the tag stack. It is 19, which is the TE label, used to forward packets to next hop 10.11.11.0 over Tunnel0.

```
PE1# show tag forwarding-table 10.11.11.11 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
21     Pop tag    10.11.11.11/32  0          Tu0       point2point
      MAC/Encaps=14/18, MTU=1500, Tag Stack{19}, via Et2/0/2
      00603E2B02410060835887428847 00013000
      No output feature configured
      Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
PE1#
```

Thus, PE1 sends a packet destined to 172.16.13.13 with the label stack {19 12308}. P1 swaps the label 19. The packet reaches P2, which pops that outer label. Then, the packet is forwarded to PE2 with only the label 12308.

On PE2, the packet with the label 12308 is received and switched according to the information in the forwarding table. This is shown here:

```
PE2# show tag for tags 12308 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
```

```

tag      tag or VC    or Tunnel Id    switched    interface
12308   Aggregate    172.16.13.0/24[V]  12256
        MAC/Encaps=0/0, MTU=0, Tag Stack{}
        VPN route: aqua
        No output feature configured
        Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
PE2#

```

**Note:** No Outgoing interface is shown because the outgoing tag is Aggregate. This is because the prefix associated with the label is the directly connected route.

Pings from CE1 to a host on CE2 confirm the VPN connectivity over the TE tunnel:

```

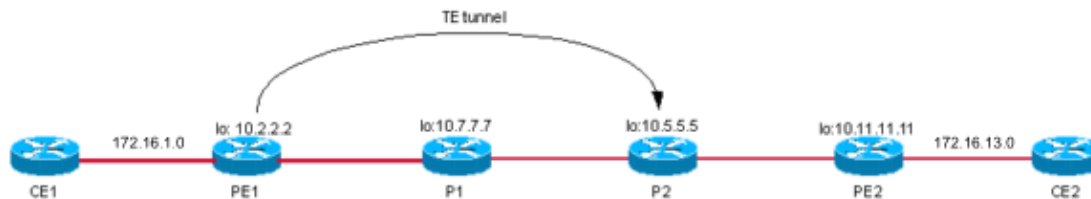
CE1# ping 172.16.13.13

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/13/36 ms
CE1#

```

## Case 2: VPN over a TE Tunnel When the TE Tunnel Is from PE1 to P2

### Topology



### Configuration

The additional TE configuration over the basic configuration on PE1 is shown here:

PE1
<pre> PE1# show run interface tunnel 0 ! interface Tunnel0  ip unnumbered Loopback0  no ip directed-broadcast  no ip route-cache distributed  tunnel destination 10.5.5.5  tunnel mode mpls traffic-eng  tunnel mpls traffic-eng autoroute announce  tunnel mpls traffic-eng path-option 10 dynamic end ! </pre>

## Verification

Check the route to prefix 172.16.13.13 on PE1 VRF aqua. It points to the next hop 10.11.11.11/32 (over Tunnel0) using label stack {19 12308}.

```
PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 11
0 packets, 0 bytes
tag information set
  local tag: VPN route head
  fast tag rewrite with Tu0, point2point, tags imposed {19 12308}
via 10.11.11.11, 0 dependencies, recursive
  next hop 10.5.5.5, Tunnel0 via 10.11.11.11/32
  valid adjacency
  tag rewrite with Tu0, point2point, tags imposed {19 12308}
PE1#
```

Label 19, the outer label, is used to reach next hop 10.11.11.11/32, as shown here:

```
PE1# show ip cef 10.11.11.11
10.11.11.11/32, version 37
0 packets, 0 bytes
tag information set
  local tag: 21
  fast tag rewrite with Tu0, point2point, tags imposed {19}
via 10.5.5.5, Tunnel0, 1 dependency
  next hop 10.5.5.5, Tunnel0
  valid adjacency
  tag rewrite with Tu0, point2point, tags imposed {19}
```

```
PE1# show mpls traffic-eng tunnels tunnel 0
```

```
Name: PE1_t0 (Tunnel0) Destination: 10.5.5.5
Status:
  Admin: up Oper: up Path: valid Signalling: connected

  path option 10, type dynamic (Basis for Setup, path weight 20)

Config Parameters:
  Bandwidth: 0 kbps (Global) Priority: 7 7 Affinity: 0x0/0xFFFF
  Metric Type: TE (default)
  AutoRoute: enabled LockDown: disabled Loadshare: 0 bw-based
  auto-bw: disabled

InLabel : -
OutLabel : Ethernet2/0/2, 19
RSVP Signalling Info:
  Src 10.2.2.2, Dst 10.5.5.5, Tun_Id 0, Tun_Instance 33
RSVP Path Info:
  My Address: 10.7.7.2
  Explicit Route: 10.7.7.7 10.8.8.7 10.8.8.5 10.5.5.5
  Record Route: NONE
  Tspec: ave rate=0 kbits, burst=1000 bytes, peak rate=0 kbits
RSVP Resv Info:
  Record Route: NONE
  Fspec: ave rate=0 kbits, burst=1000 bytes, peak rate=Inf
Shortest Unconstrained Path Info:
  Path Weight: 20 (TE)
  Explicit Route: 10.7.7.2 10.7.7.7 10.8.8.7 10.8.8.5
                  10.5.5.5

History:
Tunnel:
  Time since created: 17 hours, 31 minutes
  Time since path change: 8 minutes, 49 seconds
```

```

Current LSP:
  Uptime: 8 minutes, 49 seconds
  Selection: reoptimization
Prior LSP:
  ID: path option 10 [31]
  Removal Trigger: path verification failed
PE1#

PE1# show mpls traffic-eng tunnels tunnel 0 | i Label
  InLabel   : -
  OutLabel  : Ethernet2/0/2, 19
PE1#

```

The packet from PE1 is sent over the TE tunnel with the label stack {19 12308}. Once P1 receives the packet, it pops (PHP) the tag 19 and sends the packet with label stack {12308}. The **show** command confirms this:

```

P1> show tag for tag 19
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
19     Pop tag    10.2.2.2 0 [33]  2130      Et2/0     10.8.8.5
P1>

P1> show tag for tag 19 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
19     Pop tag    10.2.2.2 0 [33]  2257      Et2/0     10.8.8.5
      MAC/Encaps=14/14, MTU=1504, Tag Stack{
      006009E08B0300603E2B02408847
      No output feature configured
P1>

```

When P2 receives the packet with the label stack {12308}, it checks its LFIB and drops the packet because no match exists. This is the **show** command output on P2:

```

P2# show tag forwarding-table tags 12308 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
P2#

P2#
7w4d: TAG: Et0/3: recvd: CoS=0, TTL=253, Tag(s)=12308
7w4d: TAG: Et0/3: recvd: CoS=0, TTL=253, Tag(s)=12308
7w4d: TAG: Et0/3: recvd: CoS=0, TTL=253, Tag(s)=12308
7w4d: TAG: Et0/3: recvd: CoS=0, TTL=253, Tag(s)=12308
P2#
P2#

```

## Explanation

The solution to this problem is to enable TDP/LDP on the TE tunnel and to make it a tag-switched interface. In the example shown in the Solution, TDP is enabled on the Tunnel0 of PE1. P2 is configured for accepting directed hellos and forming directed TDP neighbors. So, PE1 receives the label for 10.11.11.11 from P2 via LDP. Now that Tunnel0 has been made a tag-switched interface and TDP has been enabled for the traffic to 10.11.11.11, PE1 uses both the labels; it uses the RSVP label to reach the TE tailend and the TDP label to reach 10.11.11.11.

In this scenario, PE1 uses the label stack {L2 L3 L1} to forward data to CE2 if these items are true:

- L1 is the VPN label.
- L2 is the RSVP label to reach the TE tailend.
- L3 is the TDP label to reach 10.11.11.11 (received from P2).

## Solution

The solution is to enable TDP across the TE tunnel.

## Configuration

Shown here is the TE tunnel configuration on PE1 with TDP enabled on it. The additions are in boldface.

```
PE1
PE1# show run interface tunnel 0
!
interface Tunnel0
 ip unnumbered Loopback0
 no ip directed-broadcast
 no ip route-cache distributed
 tag-switching ip

!--- This enables TDP.

 tunnel destination 10.5.5.5
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng path-option 10 dynamic
end
!
```

This is the additional configuration on the tailend of the TE tunnel to accept directed TDP hellos:

```
P2# show run | i directed-hello
tag-switching tdp discovery directed-hello accept

!--- This configures P2 to accept directed TDP hellos.

P2#
```

## Verification

```
PE1# show tag tdp neighbor | i Peer
Peer TDP Ident: 10.7.7.7:0; Local TDP Ident 10.2.2.2:0
Peer TDP Ident: 10.5.5.5:0; Local TDP Ident 10.2.2.2:0

PE1#
PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 11
0 packets, 0 bytes
tag information set
 local tag: VPN route head
 fast tag rewrite with Tu0, point2point, tags imposed {19 18 12308}
 via 10.11.11.11, 0 dependencies, recursive
 next hop 10.5.5.5, Tunnel0 via 10.11.11.11/32
 valid adjacency
 tag rewrite with Tu0, point2point, tags imposed {19 18 12308}
PE1#

PE1# show mpls traffic-eng tunnels tunnel 0 | i Label
InLabel : -
OutLabel : Ethernet2/0/2, 19

!--- This is the TE label learned via RSVP.

PE1#
PE1# show tag tdp bind 10.11.11.11 32
```

```
tib entry: 10.11.11.11/32, rev 20
  local binding: tag: 21
  remote binding: tsr: 10.7.7.7:0, tag: 17
  remote binding: tsr: 10.5.5.5:0, tag: 18
```

*!--- This is the TDP label from P2.*

When P1 receives the packet with the label stack {19 18 12308}, it pops the tag 19 and sends the packet with the label stack {18 12308} to P2. P2 checks its LFIB for label 18, then pops the tag and sends it over the outgoing interface PO2/0/0 toward PE1. PE1 receives the packet with label 12308 and switches it successfully to CE2.

```
P2# show tag for tag 18
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
18     Pop tag   10.11.11.11/32  117496     POS2/0/0  point2point
```

```
P2# show tag tdp discovery
Local TDP Identifier:
  10.5.5.5:0
Discovery Sources:
Interfaces:
  Ethernet0/3 (tdp): xmit/rcv
    TDP Id: 10.7.7.7:0
  POS2/0/0 (tdp): xmit/rcv
    TDP Id: 10.11.11.11:0
Directed Hellos:
  10.5.5.5 -> 10.2.2.2 (tdp): passive, xmit/rcv
    TDP Id: 10.2.2.2:0
```

```
P2# show tag tdp neighbor 10.2.2.2
Peer TDP Ident: 10.2.2.2:0; Local TDP Ident 10.5.5.5:0
TCP connection: 10.2.2.2.711 - 10.5.5.5.11690
State: Oper; PIEs sent/rcvd: 469/465; Downstream
Up time: 01:41:08
TDP discovery sources:
  Directed Hello 10.5.5.5 -> 10.2.2.2, passive
Addresses bound to peer TDP Ident:
  10.7.7.2          172.16.47.166    10.2.2.2
```

```
PE1# show tag tdp neighbor 10.5.5.5
Peer TDP Ident: 10.5.5.5:0; Local TDP Ident 10.2.2.2:0
TCP connection: 10.5.5.5.11690 - 10.2.2.2.711
State: Oper; PIEs sent/rcvd: 438/441; Downstream
Up time: 01:35:08
TDP discovery sources:
  Directed Hello 10.2.2.2 -> 10.5.5.5, active
```

*!--- This indicates the directed neighbor.*

```
Addresses bound to peer TDP Ident:
  10.5.5.5          10.12.12.5       10.8.8.5
```

```
PE1# show ip route 10.11.11.11
Routing entry for 10.11.11.11/32
  Known via "isis", distance 115, metric 40, type level-1
  Redistributing via isis
  B Last update from 10.5.5.5 on Tunnel0, 01:52:21 ago
  Routing Descriptor Blocks:
  * 10.5.5.5, from 10.11.11.11, via Tunnel0
    Route metric is 40, traffic share count is 1
```

A ping command from CE1 to a host on CE2 confirms the solution.

```
CE1# ping 172.16.13.13
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
```

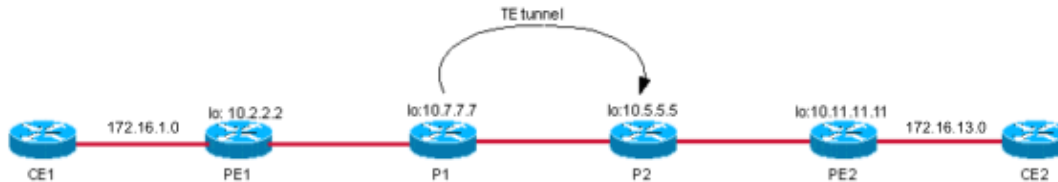
```
!!!!!
```

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

```
CE1#
```

## Case 3: VPN Between CE1 and CE2 over a TE Tunnel from P1 to P2 When TDP/LDP Is Not Enabled

### Topology



### Configuration

The tunnel configuration on PE1 is shown here:

```
PE1
P1# show run interface tunnel 0
Building configuration...

Current configuration : 255 bytes
!
interface Tunnel0
 ip unnumbered Loopback0
 no ip directed-broadcast
 ip route-cache distributed
 tunnel destination 10.5.5.5
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng path-option 10 dynamic
end
```

### Verification

Verify how packets destined to CE2 172.16.13.13 get switched here. The **show ip cef** command output shows that packets to destination 172.16.13.13 are switched with the label stack {17 12308}:

```
PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 18, cached adjacency 10.7.7.7
0 packets, 0 bytes
 tag information set
  local tag: VPN route head
  fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}
 via 10.11.11.11, 0 dependencies, recursive
  next hop 10.7.7.7, Ethernet2/0/2 via 10.11.11.11/32
  valid cached adjacency
  tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}
```

When P1 receives this packet, it removes the outer label 17 and switches the packet after looking in the IP routing table to Tunnel0. Notice the implicit-null OutLabel in this output; it means that the outgoing interface is not label switched.

```
P1# show ip cef 10.11.11.11 detail
10.11.11.11/32, version 52
0 packets, 0 bytes
tag information set
  local tag: 17
  fast tag rewrite with Tu0, point2point, tags imposed {}
via 10.5.5.5, Tunnel0, 0 dependencies
  next hop 10.5.5.5, Tunnel0
  valid adjacency
  tag rewrite with Tu0, point2point, tags imposed {}

P1# show mpls traffic-eng tunnel tunnel 0 | i Label
InLabel : -
OutLabel : Ethernet2/0, implicit-null
P1# show tag for 10.11.11.11 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
17     Untagged  10.11.11.11/32  882        Tu0        point2point
      MAC/Encaps=14/14, MTU=1500, Tag Stack{}, via Et2/0
      006009E08B0300603E2B02408847
      No output feature configured
      Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
P1# show ip route 10.11.11.11
Routing entry for 10.11.11.11/32
  Known via "isis", distance 115, metric 30, type level-1
  Redistributing via isis
  Last update from 10.5.5.5 on Tunnel0, 00:03:20 ago
  Routing Descriptor Blocks:
  * 10.5.5.5, from 10.11.11.11, via Tunnel0
    Route metric is 30, traffic share count is 1
```

Once P2 receives the packet with label 12308, it looks at its forwarding table. Because there is no way P2 can be aware of the VPN tag 12308 from CE2, it drops the packet.

```
P2# show tag for tag 12308 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
```

This breaks the path of VPN packets destined to CE2. It is confirmed by the **ping** to CE2 172.16.13.13/32.

```
PE1#
CE1# ping 172.16.13.13

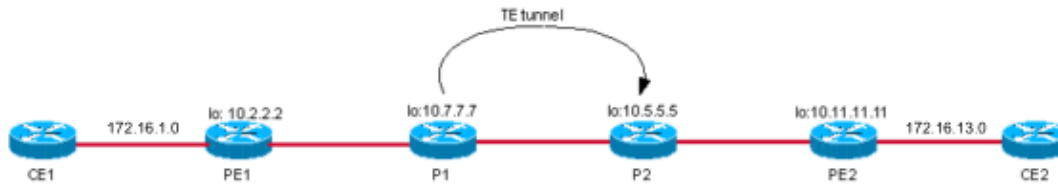
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
CE1#
```

## Solution

The solution is to enable LDP/TDP over the tunnel. The next section discusses this solution.

## Case 4: VPN over a TE Tunnel Between P1 and P2 with LDP Enabled

## Topology



## Configuration

With LDP enabled on the tunnel, the configurations on P1 appear as shown here. Additions are in boldface.

```
PE1
P1# show run interface tunnel 0
Building configuration...

Current configuration : 273 bytes
!
interface Tunnel0
 ip unnumbered Loopback0
 no ip directed-broadcast
 ip route-cache distributed
 mpls label protocol ldp
 tunnel destination 10.5.5.5
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng path-option 10 dynamic
end
!
```

## Verification

PE1 sends packets to prefix 172.16.13.13/32 with the label stack {17 12308}.

```
PE1#
PE1# show tag for 10.11.11.11 detail
Local   Outgoing   Prefix           Bytes tag   Outgoing   Next Hop
tag     tag or VC  or Tunnel Id     switched   interface
21      17         10.11.11.11/32   0          Et2/0/2    10.7.7.7
        MAC/Encaps=14/18, MTU=1500, Tag Stack{17}
        00603E2B02410060835887428847 00011000
        No output feature configured
        Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

PE1#
PE1# show ip cef 10.11.11.11 detail
10.11.11.11/32, version 60, cached adjacency 10.7.7.7
0 packets, 0 bytes
tag information set
  local tag: 21
  fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17}
  via 10.7.7.7, Ethernet2/0/2, 1 dependency
  next hop 10.7.7.7, Ethernet2/0/2
  valid cached adjacency
  tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17}

PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 18, cached adjacency 10.7.7.7
```

```

0 packets, 0 bytes
tag information set
  local tag: VPN route head
  fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}
via 10.11.11.11, 0 dependencies, recursive
  next hop 10.7.7.7, Ethernet2/0/2 via 10.11.11.11/32
  valid cached adjacency
  tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}

```

P1 receives the packet with label stack {17 12308} and looks at its LFIB for label 17.

```

P1# show tag for tag 17 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
17     18         10.11.11.11/32  1158      Tu0       point2point
      MAC/Encaps=14/18, MTU=1496, Tag Stack{18}, via Et2/0
      006009E08B0300603E2B02408847 00012000
      No output feature configured
      Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
P1#

```

```

P1# show ip cef 10.11.11.11 detail
10.11.11.11/32, version 52
0 packets, 0 bytes
tag information set
  local tag: 17
  fast tag rewrite with Tu0, point2point, tags imposed {18}
via 10.5.5.5, Tunnel0, 0 dependencies
  next hop 10.5.5.5, Tunnel0
  valid adjacency
  tag rewrite with Tu0, point2point, tags imposed {18}

```

It shows that label 17 should be swapped to label 18. Therefore, that packet is switched over the tunnel interface with the label stack {18 12308}.

P2 receives the packet over its tunnel interface with label stack {18 12308}. It pops the tag 18 (because it is the penultimate hop router) and switches the packet to PE2 with the label 12308.

```

P2# show tag for tag 18 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
18     Pop tag    10.11.11.11/32  127645    PO2/0/0   point2point
      MAC/Encaps=4/4, MTU=4474, Tag Stack{
      0F008847
      No output feature configured
      Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
P2#

```

PE2 receives the packet with label 12308, which switches the packet to CE2 successfully.

```

PE2# show tag forwarding tags 12308 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
tag    tag or VC  or Tunnel Id    switched   interface
12308  Aggregate  172.16.13.0/24[V]  12256
      MAC/Encaps=0/0, MTU=0, Tag Stack{
      VPN route: aqua
      No output feature configured
      Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
PE2#

```

```

CE1# ping 172.16.13.13

```

Type escape sequence to abort.

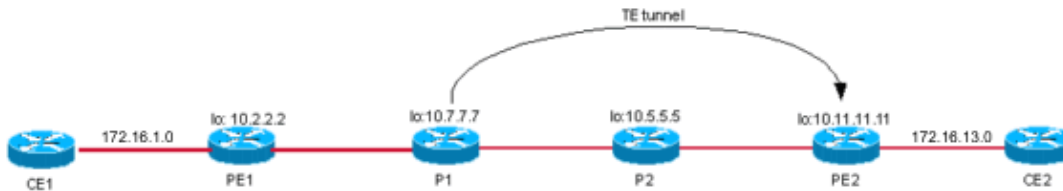
```

Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
CE1#

```

## Case 5: MPLS VPN over a Tunnel Between P1 and PE2

### Topology



### Configuration

PE1
<pre> P1# show run interface tunnel 0 Building configuration...  Current configuration : 258 bytes ! interface Tunnel0  ip unnumbered Loopback0  no ip directed-broadcast  ip route-cache distributed  tunnel destination 10.11.11.11  tunnel mode mpls traffic-eng  tunnel mpls traffic-eng autoroute announce  tunnel mpls traffic-eng path-option 10 dynamic end </pre>

### Verification

PE1 sends a packet destined to 172.16.13.13 to its next hop 10.11.11.11 with the label stack {17 12308}.

```

PE1# show ip cef vrf aqua 172.16.13.13
172.16.13.0/24, version 18, cached adjacency 10.7.7.7
0 packets, 0 bytes
 tag information set
   local tag: VPN route head
   fast tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}
 via 10.11.11.11, 0 dependencies, recursive
   next hop 10.7.7.7, Ethernet2/0/2 via 10.11.11.11/32
   valid cached adjacency
   tag rewrite with Et2/0/2, 10.7.7.7, tags imposed {17 12308}

```

P1 receives the packet with label stack {17 12308}. P1 looks at its LFIB table and checks the tag stack {17} and switches the packet with label {17} toward P2.

```

P1# show tag for 10.11.11.11 detail
Local  Outgoing  Prefix          Bytes tag  Outgoing   Next Hop
tag    tag or VC  or Tunnel Id   switched  interface
17     Untagged  10.11.11.11/32 411        Tu0        point2point
      MAC/Encaps=14/18, MTU=1500, Tag Stack{17}, via Et2/0

```

```
006009E08B0300603E2B02408847 00011000
No output feature configured
Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

```
P1# show tag for tag 17 detail
```

```
Local Outgoing Prefix Bytes tag Outgoing Next Hop
tag tag or VC or Tunnel Id switched interface
17 Untagged 10.11.11.11/32 685 Tu0 point2point
MAC/Encaps=14/18, MTU=1500, Tag Stack{17}, via Et2/0
006009E08B0300603E2B02408847 00011000
No output feature configured
Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
P1#
```

```
P1# show ip cef 10.11.11.11
```

```
10.11.11.11/32, version 67
0 packets, 0 bytes
tag information set
local tag: 17
fast tag rewrite with Tu0, point2point, tags imposed {17}
via 10.11.11.11, Tunnel0, 0 dependencies
next hop 10.11.11.11, Tunnel0
valid adjacency
tag rewrite with Tu0, point2point, tags imposed {17}
```

P2 receives the packet with label stack {17 12308}. P2, being the penultimate hop router, pops label 17.

```
P2# show tag for tag 17 detail
```

```
Local Outgoing Prefix Bytes tag Outgoing Next Hop
tag tag or VC or Tunnel Id switched interface
17 Pop tag 10.7.7.7 0 [5] 535 PO2/0/0 point2point
MAC/Encaps=4/4, MTU=4474, Tag Stack{}
0F008847
No output feature configured
P2#
```

PE2 then receives the packet with the label 12308. PE2 is aware that the destination for label 12308 is directly connected. Therefore, the **ping** from CE1 to CE2 is 10.

```
PE2# show tag for tag 12308 detail
```

```
Local Outgoing Prefix Bytes tag Outgoing Next Hop
tag tag or VC or Tunnel Id switched interface
12308 Aggregate 172.16.13.0/24[V] 12776
MAC/Encaps=0/0, MTU=0, Tag Stack{}
VPN route: aqua
No output feature configured
Per-packet load-sharing, slots: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
PE2#
```

**Note:** No Outgoing interface is shown because the outgoing tag is Aggregate. This is because the prefix associated with the label is the directly connected route.

```
CE1# ping 172.16.13.13
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.13, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
CE1#
```

## Known Issues

Refer to Field Notice: MPLS VPN with TE and MPLS InterAS Advisory on Cisco IOS® Software for more details.

## Conclusion

When the TE tunnel is terminated on the egress PE, the MPLS VPN and the TE work together without any additional configuration. When the TE tunnel is terminated on any P routers (before the PE in the core), the MPLS VPN traffic forwarding fails because packets arrive with VPN labels as the outer labels, which are not in the LFIBs of these devices. Therefore, these intermediate routers are not able to forward packets to the final destination, the VPN customer network. In such a case, LDP/TDP should be enabled on the TE tunnel to solve the problem.

---

## Related Information

- [MPLS FAQ For Beginners](#)
- [How to Troubleshoot the MPLS VPN](#)
- [MPLS Basic Traffic Engineering Using OSPF Configuration Example](#)
- [Configuring a Basic MPLS VPN](#)
- [Troubleshooting LSP Failure in MPLS VPN](#)
- [MPLS Support Page](#)
- [Technical Support – Cisco Systems](#)

---

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2008 – 2009 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

---

Updated: Aug 10, 2005

Document ID: 29828

---