

Chapter 14

Translational Cross-Connect and Layer 2.5 VPNs

Translational cross-connect (TCC) is a switching concept that allows you to establish interconnections between a variety of Layer 2 protocols or circuits. It is similar to its predecessor, circuit cross-connect (CCC). However, while CCC requires the same Layer 2 encapsulations on both sides of a Juniper Networks router (such as PPP-to-PPP or Frame Relay-to-Frame Relay), TCC lets a network administrator connect different types of Layer 2 protocols interchangeably. With TCC, combinations such as PPP-to-ATM and Ethernet-to-Frame Relay cross-connections are possible.

This feature guide covers these topics:

Overview on page 464

System Requirements on page 464

Terms and Acronyms on page 465

Configure TCC Interface Switching on page 465

Example: PPP to ATM TCC Configuration on page 470

Example: Frame Relay to Fast Ethernet TCC Configuration on page 472

Configure Layer 2.5 VPNs on page 474

Example: Layer 2.5 VPN Configuration on page 476

Check Your Work on page 482

For More Information on page 487

Revision History on page 487

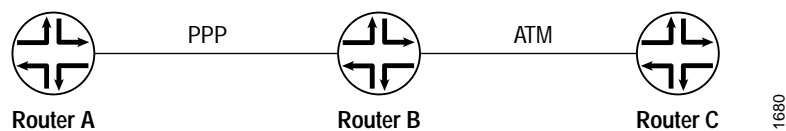
Overview

The JUNOS software makes interworking between unlike protocols possible. The software strips off the Layer 2 header when frames enter the router and adds a different Layer 2 header before the frames exit the router. TCC supports these Layer 2 protocols:

- ATM
- Cisco HDLC
- Ethernet
- Extended VLAN
- Frame Relay
- PPP

In Figure 48, the PPP header is stripped from frames arriving at Router B and an ATM header is added before the frames are sent to Router C. All Layer 2 negotiations are terminated at the interconnecting router (Router B). Examples include Link Control Protocol (LCP) and Network Control Protocol (NCP) for PPP, keepalives for Cisco HDLC, and Local Management Interface (LMI) for Frame Relay.

Figure 48: TCC Concept Example



TCC functionality is different from standard Layer 2 switching. TCC only swaps Layer 2 headers. No other processing, such as header checksums, time-to-live (TTL) decrementing, or protocol handling, is performed. Currently, TCC is supported in IPv4.

This guide shows you how to use the Layer 2 interworking nature of TCC for interface switching and for constructing Layer 2.5 virtual private networks (VPNs).

System Requirements

To implement TCC, your system must meet these minimum requirements:

- JUNOS Release 6.0 or later for CCC, TCC, and Layer 2.5 VPN support on T-series platforms

- JUNOS Release 5.6 or later for proxy Address Resolution Protocol (ARP) functionality in Ethernet TCC and extended VLAN TCC networks on M-series routers

- JUNOS Release 5.4 or later for Ethernet TCC and extended VLAN TCC interface encapsulation types on M-series routers

JUNOS Release 5.2 or later for PPP TCC, Cisco HDLC TCC, ATM TCC, and Frame Relay TCC interface encapsulation types on M-series routers

Any Gigabit Ethernet or Fast Ethernet PIC for Ethernet TCC and extended VLAN TCC interface encapsulation types (except the Fast Ethernet 48-port PIC)

Any combination of three Juniper Networks M-series or T-series Internet platforms for TCC or five platforms for Layer 2.5 VPNs.

Terms and Acronyms

Circuit cross-connect (CCC)—A method of exchanging frames between two router interfaces running the same Layer 2 protocol, such as ATM, Cisco HDLC, Ethernet, extended VLAN, Frame Relay, and PPP. For more information about CCC, see the *JUNOS Internet Software Configuration Guide: Network Interfaces and Class of Service*.

Translational cross-connect (TCC)—A Juniper Networks method of exchanging frames between two router interfaces running different Layer 2 protocols, such as ATM, Cisco HDLC, Ethernet, extended VLAN, Frame Relay, and PPP. For more information about TCC, see the *JUNOS Internet Software Configuration Guide: Network Interfaces and Class of Service*.

Layer 2.5 VPNs—A method of privately connecting sites across the Internet. Layer 2.5 VPNs function like Layer 2 VPNs, but connect two sites that use different Layer 2 protocols. For more information about Layer 2 VPNs, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Provider Edge router (PE)—Any router that connects customer edge (CE) routers to the provider core network.

Customer Edge router (CE)—Any router that connects a customer site to a PE router.

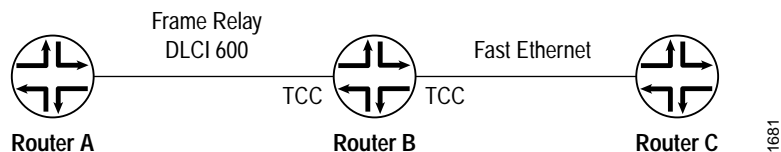
Provider core router (P)—Any router that lies between PE routers in the provider core network.

Configure TCC Interface Switching

TCC joins disparate Layer 2 logical interfaces by means of a virtual Layer 2 switching technique.

Figure 49 illustrates Layer 2 TCC switching. In this example, Router A is connected by Frame Relay to Router B and Router B is connected by Fast Ethernet to Router C. Router B is a Juniper Networks router. TCC allows you to configure Router B to act as a multiservice Layer 2 switch. To do this, you configure a connection from Router A to Router C that passes through Router B. This effectively establishes Router B as a Frame Relay-to-Fast Ethernet switch and allows the router to switch frames transparently between Router A and Router C without regard to the frames' contents or the Layer 3 protocols. Router B removes the Frame Relay header from frames arriving on data-link connection identifier (DLCI) 600 and adds an Ethernet header onto the frames before sending them to the Fast Ethernet link.

Figure 49: Layer 2 TCC Switching



You can configure Layer 2 TCC switching on ATM, Cisco HDLC, Ethernet, extended VLAN, Frame Relay, and PPP circuits. TCC enables unlike interfaces to be connected with a single cross-connect.

To configure Layer 2 TCC switching, perform the following tasks on the router that is acting as the switch (Router B in Figure 49):

Define the Encapsulation for Layer 2 TCC Switching on page 466

Define the Connection for Layer 2 TCC Switching on page 469

Configure MPLS on page 470

To view examples of Layer 2 TCC switching, see the following sections:

Example: PPP to ATM TCC Configuration on page 470

Check Your Work on page 472

Example: Frame Relay to Fast Ethernet TCC Configuration on page 472

Check Your Work on page 474

Define the Encapsulation for Layer 2 TCC Switching

To begin implementation of Layer 2 TCC switching, configure TCC encapsulation on the desired interfaces of the router that is acting as the switch (Router B in Figure 49).



Issue

You cannot configure standard protocol families on TCC or CCC interfaces. Only the CCC family is allowed on CCC encapsulated interfaces. Likewise, only the TCC family is allowed on TCC-encapsulated interfaces.

For ATM connections, specify the encapsulation type when configuring ATM virtual circuits (VCs) at the [edit interfaces *interface-name* unit *unit-number*] hierarchy level. For each VC, you configure whether it is a circuit or a regular logical interface. The default interface type is point-to-point.

```
[edit]
interfaces {
  at-fpc/pic/port {
    atm-options {
      vpi vpi-identifier maximum-vcs maximum-vcs;
    }
    unit logical-unit-number {
      point-to-point;      # Default interface type
      encapsulation (atm-tcc-vc-mux | atm-tcc-snap);
      vci vpi-identifier.vci-identifier;
    }
  }
}
```

For Cisco HDLC and PPP circuits, specify the encapsulation in the encapsulation statement at the [edit interfaces *interface-name*] hierarchy level. This statement configures the entire physical device. Also, you must configure the logical interface unit 0.

```
[edit]
interfaces {
  type-fpc/pic/port {
    encapsulation (cisco-hdlc-tcc | ppp-tcc);
    unit 0;
  }
}
```

You can specify the encapsulation for Frame Relay circuits at the [edit interfaces *interface-name*] hierarchy level and the [edit interfaces *interface-name* unit *unit-number*] hierarchy level. For TCC and CCC interfaces, the DLCI value must be configured in the range of 512 through 1022. The default interface type is point-to-point.

```
[edit]
interfaces {
  type-fpc/pic/port {
    encapsulation frame-relay-tcc;
    unit logical-unit-number {
      point-to-point;      # Default interface type
      encapsulation frame-relay-tcc;
      dlci dlci-identifier;
    }
  }
}
```

Configure Ethernet Encapsulation with Remote and Proxy ARP Addresses

For Ethernet TCC circuits, specify the encapsulation with the encapsulation ethernet-tcc statement. This statement configures the entire physical device.

To provide Address Resolution Protocol (ARP) functionality for an Ethernet-based neighbor, configure the remote statement at the [edit interfaces *interface-name* unit *unit-number* family tcc] hierarchy level and specify either the MAC address or IP address of the TCC router's Ethernet neighbor. To complete the setup of ARP, configure the proxy statement at the [edit interfaces *interface-name* unit *unit-number* family tcc] hierarchy level and specify the IP address of the TCC router's non-Ethernet neighbor.

```
[edit]
interfaces
  EthernetType-fpc/pic/port {
    encapsulation ethernet-tcc;
    unit 0 {
      family tcc {
        remote {          # Addresses associated with the Ethernet TCC neighbor.
          mac-address mac-address; # Select either a MAC address or an IP address.
          inet-address inet-address;
        }
        proxy {          # Addresses belonging to the non-Ethernet TCC neighbor.
          inet-address inet-address;
        }
      }
    }
  }
}
```

Configure Extended VLAN Encapsulation with Remote and Proxy ARP Addresses

Specify the encapsulation for extended VLAN circuits with the encapsulation extended-vlan-tcc statement. This statement configures the entire physical device.

You must also enable VLAN tagging. Ethernet interfaces in VLAN mode can have multiple logical interfaces. For encapsulation type extended-vlan-tcc, all VLAN IDs from 0 through 4094 are valid, up to a maximum of 1024 VLANs.

To enable ARP functionality, configure the remote statement at the [edit interfaces *interface-name* unit *unit-number* family tcc] hierarchy level with either the MAC address or IP address of your Ethernet TCC neighbor. To complete the ARP setup, configure the proxy statement at the [edit interfaces *interface-name* unit *unit-number* family tcc] hierarchy level and specify the IP address of the non-Ethernet TCC neighbor.

```
[edit]
interfaces {
  EthernetType-fpc/pic/port {
    vlan-tagging;
    encapsulation extended-vlan-tcc;
    unit 0 {
      vlan-id 600;
      family tcc {
        remote {          # Addresses associated with the Ethernet TCC neighbor.
          mac-address mac-address; # Select either a MAC address or an IP address.
          inet-address inet-address;
        }
        proxy {          # Addresses belonging to the non-Ethernet TCC neighbor.
          inet-address inet-address;
        }
      }
    }
  }
}
```

Option: Configure Static ARP on the Ethernet Neighbor Instead of Proxy ARP

If you do not use the proxy statement in the Ethernet TCC and extended VLAN TCC encapsulation hierarchies shown earlier, you must use another method to allow ARP to continue to function. To retain the functionality of ARP for Ethernet networks, you must configure static ARP on the Ethernet neighbor. Use of static ARP assumes that you have already configured the remote statement on the TCC router (see “Configure Ethernet Encapsulation with Remote and Proxy ARP Addresses” on page 467 and “Configure Extended VLAN Encapsulation with Remote and Proxy ARP Addresses” on page 468).

You configure the arp statement on the Ethernet neighbor at the [edit interfaces *interface-number* unit *unit-number* family inet address *ip-address*] hierarchy level. Your static ARP statement must contain the IP address of the non-Ethernet neighbor on the opposite side of the TCC router and the Ethernet interface MAC address of the TCC router. This static ARP configuration enables return path ARP functionality and complements the remote statement previously set on the TCC router.

In Figure 51 on page 472, you would configure an ARP statement on the fe-0/0/0 interface of Router C. The ARP statement would contain the IP address for interface so-0/1/0.600 on Router A and the MAC address of the fe-1/0/0 interface of Router B.

Configure static ARP on an Ethernet neighbor at the [edit interfaces *interface-name* unit *unit-number* family inet address *ip-address*] hierarchy level.

```
[edit]
interfaces
  EthernetType-fpc/pic/port {
    unit 0 {
      family inet {
        address ip-address {           # The local IP Address
          arp ip-address mac mac-address; # IP address of the non-Ethernet TCC
        }                               # neighbor and MAC Address of the TCC
      }                                 # router's Ethernet interface
    }
  }
}
```

Define the Connection for Layer 2 TCC Switching

The next step in configuring Layer 2 TCC switching is to define the connection between the two circuits. You configure this on the router acting as the TCC switch. When you specify the interface names, include the logical portion of the name, which corresponds to the logical unit number. The cross-connect is bidirectional, so packets received on the first interface are transmitted by the second interface, and those received on the second interface are transmitted by the first.

```
[edit]
protocols {
  connections {
    interface-switch connection-name {
      interface first-interface-name.unit-number;
      interface second-interface-name.unit-number;
    }
  }
}
```

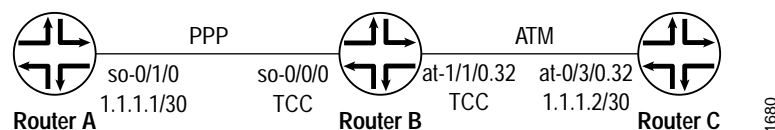
Configure MPLS

You must also configure MPLS for a Layer 2 cross-connect to work. The following is a minimal MPLS configuration:

```
[edit]
interfaces {
  interface-name {
    unit logical-unit-number;
  }
}
protocols {
  mpls {
    interface all;
  }
}
```

Example: PPP to ATM TCC Configuration

Figure 50: TCC Interface Switching—PPP to ATM



In Figure 50, Router A uses PPP to connect with Router B, while Router C connects with Router B through ATM. Router B acts as the Layer 2 virtual switch and transparently connects Router A to Router C.

On Router A, configure basic PPP encapsulation and any desired Layer 3 protocol families on the SONET interface.

```
Router A [edit]
interfaces {
  so-0/1/0 {
    description "to Router B so-0/0/0";
    unit 0 {
      encapsulation ppp;
      family inet {
        address 1.1.1.1/30;
      }
    }
  }
}
```

Router B acts as the virtual Layer 2 switch. Here you configure the appropriate TCC encapsulations on the corresponding interfaces. In this case, encapsulation ppp-tcc is bound to physical interface so-0/0/0, and encapsulation atm-tcc-vc-mux is placed on VC 32 of interface at-1/1/0. Because the switching occurs at Layer 2, you cannot configure IP addresses or other Layer 3 family information on these interfaces.

You also need to configure MPLS and establish the cross-connect by adding the necessary interfaces to the interface-switch statement at the [edit protocols connections] hierarchy level.

```

Router B [edit]
interfaces {
  so-0/0/0 {
    description "to Router A so-0/1/0";
    encapsulation ppp-tcc;
    unit 0 {
    }
  }
  at-1/1/0 {
    description "to Router C at-0/3/0";
    atm-options {
      vpi 0 maximum-vcs 2000;
    }
    unit 32 {
      vci 32;
      encapsulation atm-tcc-vc-mux;
    }
  }
}
protocols {
  mpls {
    interface so-0/0/0.0;
    interface at-1/1/0.32;
  }
  connections {
    interface-switch PPP-to-ATM {
      interface so-0/0/0.0;
      interface at-1/1/0.32;
    }
  }
}

```

On Router C, the encapsulation option used to connect to the TCC-encapsulated ATM interface on Router B is atm-vc-mux. Since this ATM connection is switched at Layer 2 to reach the PPP link, it is transparent to Layer 3 addressing. As a result, the IP address must be configured in the same address space as Router A's so-0/1/0 interface.

```

Router C [edit]
interfaces {
  at-0/3/0 {
    description "to Router B at-1/1/0";
    atm-options {
      vpi 0 maximum-vcs 2000;
    }
    unit 32 {
      vci 32;
      encapsulation atm-vc-mux;
      family inet {
        address 1.1.1.2/30;
      }
    }
  }
}

```

Check Your Work

To verify your TCC connection, use the show connections command on Router B:

```

user@router_b> show connections
CCC and TCC connections [Link Monitoring On]
Legend for status (St)
UN -- uninitialized
NP -- not present
WE -- wrong encapsulation
DS -- disabled
Dn -- down
-> -- only outbound conn is up
<- -- only inbound conn is up
Up -- operational

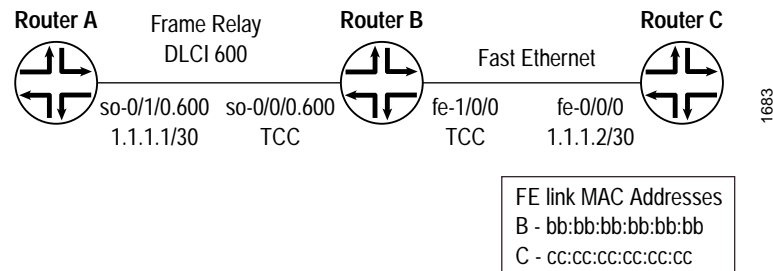
Legend for connection types
if-sw: interface switching
rmt-if: remote interface switching
lsp-sw: LSP switching

Legend for circuit types
intf -- interface
tlsp -- transmit LSP
rlsp -- receive LSP

Connection/Circuit      Type  St  Time last up  # Up trans
PPP-to-ATM
at-1/1/0.32            intf  Up  Nov 30 08:57:53  1
so-0/0/0.0             intf  Up
    
```

Example: Frame Relay to Fast Ethernet TCC Configuration

Figure 51: TCC Interface Switching—Frame Relay to Fast Ethernet



In the configuration example in Figure 51, Router A uses Frame Relay to connect with Router B, while Router C connects to Router B by using Fast Ethernet. Router B acts as the Layer 2 virtual switch and transparently connects Router A to Router C.

You must enable Frame Relay encapsulation on Router A at the physical interface level.

```

Router A [edit]
interfaces {
  so-0/1/0 {
    description "to Router B so-0/0/0";
    encapsulation frame-relay;
    unit 600 {
      point-to-point;
      dlci 600;
      family inet {
        address 1.1.1.1/30;
      }
    }
  }
}
    
```

Router B acts as the virtual switch. Enable the appropriate TCC encapsulations on the corresponding interfaces. In this case, configure the encapsulation `frame-relay-tcc` option on the logical and physical interfaces of `so-0/0/0.600`. Next, add the `ethernet-tcc` encapsulation type to the physical interface of `fe-1/0/0`. To enable ARP, configure the remote MAC address or IP address of Router C's Fast Ethernet interface with the `remote` statement at the `[edit interfaces interface-name unit 0 family tcc]` hierarchy level. To enable proxy ARP, include the `proxy` statement at the `[edit interfaces interface-name unit 0 family tcc]` hierarchy level and specify the IP address of Router A.

After configuring the correct interface encapsulations, complete your cross-connect by adding both interfaces into your MPLS configuration. Include the same interfaces in the `interface-switch` statement at the `[edit protocols connections]` hierarchy level.

```
Router B [edit]
interfaces {
  so-0/0/0 {
    description "to Router A so-0/1/0";
    dce;
    encapsulation frame-relay-tcc;
    unit 600 {
      point-to-point;
      encapsulation frame-relay-tcc;
      dlci 600;
    }
  }
  fe-1/0/0 {
    description "to Router C fe-0/0/0";
    encapsulation ethernet-tcc;
    unit 0 {
      family tcc {
        remote { # Addresses associated with the Ethernet TCC neighbor—Router C.
          mac-address cc:cc:cc:cc:cc:cc; # Or, specify Router C's IP address 1.1.1.2 here.
        }
        proxy { # Addresses associated with the non-Ethernet TCC neighbor—Router A.
          inet-address 1.1.1.1
        }
      }
    }
  }
}
protocols {
  mpls {
    interface so-0/0/0.600;
    interface fe-1/0/0.0;
  }
  connections {
    interface-switch FR-to-Ether {
      interface so-0/0/0.600;
      interface fe-1/0/0.0;
    }
  }
}
```

Ethernet encapsulation is the default for Router C. Because the Fast Ethernet connection is switched at Layer 2 to reach the Frame Relay link, it is transparent to Layer 3 addressing. As a result, you must configure the IP address for the `fe-0/0/0` interface in the same address space as Router A's `so-0/1/0.600` interface.

Optionally, configure static ARP on the fe-0/0/0 interface if you omit the proxy statement on Router B. The arp statement must contain the IP address from interface so-0/1/0.600 on Router A and the MAC address of the Fast Ethernet interface on Router B.

```
Router C [edit]
interfaces
  fe-0/0/0 {
    description "to Router B fe-1/0/0";
    unit 0 {
      family inet {
        address 1.1.1.2/30 {
          arp 1.1.1.1 mac bb:bb:bb:bb:bb:bb; # Configure this only if you did not enter a proxy
                                             # statement on Router B.
        }
      }
    }
  }
}
```

Check Your Work

To verify the operational status of your TCC connection, use the show connections command on Router B:

```
user@router_b> show connections
CCC and TCC connections [Link Monitoring On]
Legend for status (St)           Legend for connection types
UN -- uninitialized             if-sw: interface switching
NP -- not present               rmt-if: remote interface switching
WE -- wrong encapsulation       lsp-sw: LSP switching
DS -- disabled
Dn -- down
-> -- only outbound conn is up   Legend for circuit types
<- -- only inbound conn is up   intf -- interface
Up -- operational               tlsp -- transmit LSP
                                rlsp -- receive LSP

Connection/Circuit              Type   St   Time last up   # Up trans
FR-to-Ether                     if-sw  Up   Dec 30 09:57:23   1
  so-0/0/0.600                  intf   Up
  fe-1/0/0.0                    intf   Up
```

Configure Layer 2.5 VPNs

Layer 2.5 VPNs are an extension to Layer 2 VPNs. The main difference is that Layer 2.5 VPNs are not required to have the same media on both ends of the connection. Layer 2.5 VPNs have the same capabilities as the CCC and Layer 2 VPN implementation in JUNOS 5.2 and later.

Layer 2.5 VPNs support the same media types as TCC: ATM, Cisco HDLC, Ethernet, extended VLAN, Frame Relay, and PPP. The only traffic type that is currently supported is IPv4. Also, IS-IS adjacencies cannot be formed across a TCC connection, because they use Open Systems Interconnection (OSI) framing.

To configure Layer 2.5 VPNs, complete the following tasks:

Configure the Encapsulation on Interfaces Participating in the Layer 2.5 VPN on page 475

Configure the Layer 2.5 VPN on page 476

To view an example of a Layer 2.5 VPN, go to the following sections:

Example: Layer 2.5 VPN Configuration on page 476

Check Your Work on page 482

Configure the Encapsulation on Interfaces Participating in the Layer 2.5 VPN

The encapsulation types used for Layer 2.5 VPNs are parallel to CCC encapsulations and identical to the TCC encapsulations explained in “Configure TCC Interface Switching” on page 465. In summary, the encapsulation types are:

atm-tcc-vc-mux

atm-tcc-snap

cisco-hdlc-tcc

ethernet-tcc

extended-vlan-tcc

frame-relay-tcc

ppp-tcc

When you configure a TCC encapsulation type, some modifications are needed to handle VPN connections over unlike Layer 2 links and to terminate the Layer 2 protocol locally. The router performs the following media-specific changes:

PPP TCC—Both Link Control Protocol (LCP) and Network Control Protocol (NCP) are terminated on the router. Internet Protocol Control Protocol (IPCP) IP address negotiation is not supported. The JUNOS software strips all PPP encapsulation data from incoming frames before forwarding them. For frames destined to a PPP-connected neighbor, PPP encapsulation is added.

Cisco HDLC TCC—Keepalive processing is terminated on the router. The JUNOS software strips all Cisco HDLC encapsulation data from incoming frames before forwarding them. Cisco-HDLC encapsulation is added onto frames that are destined to a Cisco HDLC-connected neighbor.

Frame Relay TCC—All Local Management Interface (LMI) processing is terminated on the router. The JUNOS software strips all Frame Relay encapsulation data from incoming frames before forwarding them. For frames destined to a Frame Relay-connected neighbor, Frame Relay encapsulation is added.

ATM—Operation, Administration, and Maintenance (OAM) and Interim Local Management Interface (ILMI) processing is terminated at the router. Cell relay is not supported. The JUNOS software strips all ATM encapsulation data from incoming frames before forwarding them. ATM encapsulation is added onto frames that are destined to an ATM-connected neighbor.

Configure the Layer 2.5 VPN

Layer 2.5 VPNs use essentially the same configuration format as Layer 2 VPNs. You should already have experience configuring Layer 2 VPNs before you attempt to configure a Layer 2.5 VPN. The major steps required to create a Layer 2 VPN are:

Enable MPLS on interfaces pointing toward the core and the edge on your provider edge (PE) and provider core (P) routers.

Configure Label Distribution Protocol (LDP) on all P and PE routers for traffic traveling from the PEs, through the core, and to the remote PEs.

Establish an internal BGP (IBGP) Layer 2 VPN peering relationship between PE routers.

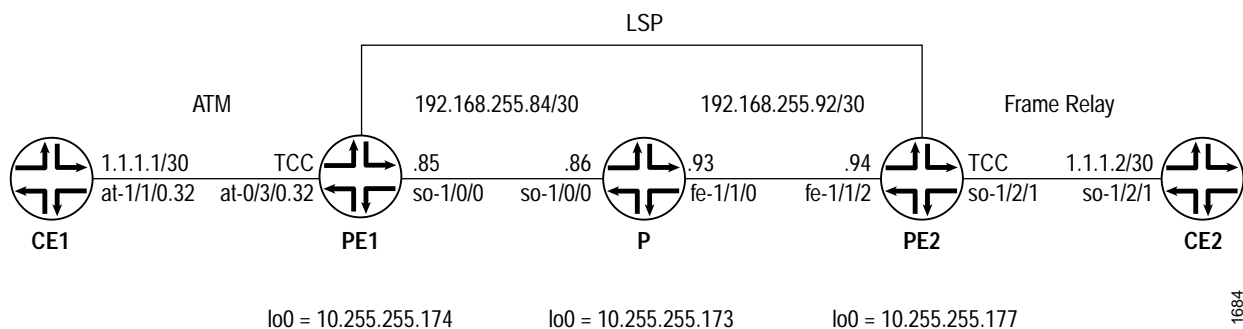
Set up policies on your PE routers that will set a private community tag on outbound BGP traffic heading to the remote PEs and accept incoming traffic that matches similar community traffic from the remote PEs.

Build VPN routing and forwarding (VRF) instances on your PE routers and apply the previously configured policies to deliver private traffic to the customer edge (CE) routers.

For a full explanation of Layer 2 VPNs and configuration samples, see the *JUNOS Internet Software Configuration Guide: VPNs*.

Example: Layer 2.5 VPN Configuration

Figure 52: Layer 2.5 VPN Topology Diagram



1684

In Figure 52 on page 476, ATM is configured between CE1 and PE1 and Frame Relay is configured between PE2 and CE2. To begin the Layer 2 VPN configuration, enable ATM and the corresponding encapsulation on CE1.

```
CE1 [edit]
interfaces
  at-1/1/0 {
    description "to PE1 at-0/3/0";
    atm-options {
      vpi 0 maximum-vc 2000;
    }
    unit 32 {
      vci 32;
      encapsulation atm-vc-mux;
      family inet {
        address 1.1.1.1/30;
      }
    }
  }
}
```

The first provider edge (PE1) router uses ATM TCC encapsulation on the ATM VC connecting to CE1. After this, standard Layer 2 VPN design rules apply. You use MPLS on interfaces pointing toward the core and the edge, establish a Layer 2 VPN BGP peer relationship with PE2, use LDP or Resource Reservation Protocol (RSVP) for traffic traveling through the core, and configure the proper VRF instance. Finally, you create policies for PE1 that will set a private community tag on outbound BGP traffic heading to PE2 and accept incoming traffic that matches similar community traffic from PE2.

```
PE1 [edit]
interfaces {
  at-0/3/0 {
    description "to CE1 at-1/1/0";
    atm-options {
      vpi 0 maximum-vc 2000;
    }
    unit 32 {
      encapsulation atm-tcc-vc-mux;
      vci 32;
    }
  }
  so-1/0/0 {
    description "to P so-1/0/0";
    unit 0 {
      family inet {
        address 192.168.255.86/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.255.174/32;
      }
    }
  }
}
```

```

protocols {
  mpls {
    interface at-0/3/0.32;
    interface so-1/0/0.0;
  }
  bgp {
    group my-internal-peers {
      type internal;
      local-address 10.255.255.174;
      family I2-vpn {
        unicast;
      }
      neighbor 10.255.255.177;
    }
  }
  ldp {
    interface so-1/0/0.0;
  }
}
policy-options {
  policy-statement companyA-import {
    term T1 {
      from {
        protocol bgp;
        community companyA;
      }
      then accept;
    }
    term Final {
      then reject;
    }
  }
  policy-statement companyA-export {
    term T1 {
      then {
        community add companyA;
        accept;
      }
    }
    term Final {
      then reject;
    }
  }
  community companyA members target:100:1;
}

```

```

routing-instances {
  companyA {
    instance-type l2vpn;
    interface at-0/3/0.32;
    route-distinguisher 10.255.255.174:1;
    vrf-import companyA-import;
    vrf-export companyA-export;
    protocols {
      l2vpn {
        encapsulation-type interworking;
        site Denver {
          site-identifier 1;
          interface at-0/3/0.32 {
            remote-site-id 2;
          }
        }
      }
    }
  }
}

```

On the provider core router (P), you need only enable MPLS and LDP on the interfaces that bridge the gap between the PE routers.

```

P [edit]
interfaces {
  so-1/0/0 {
    description "to PE1 so-1/0/0";
    unit 0 {
      family inet {
        address 192.168.255.85/30;
      }
      family mpls;
    }
  }
  fe-1/1/0 {
    description "to PE2 fe-1/1/2";
    unit 0 {
      family inet {
        address 192.168.255.93/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.255.173/32;
      }
    }
  }
}

```

```

protocols {
  mpls {
    interface so-1/0/0.0;
    interface fe-1/1/0.0;
  }
  ldp {
    interface so-1/0/0.0;
    interface fe-1/1/0.0;
  }
}

```

The PE2 router uses Frame Relay TCC encapsulation on the Frame Relay DLCI connecting to CE2. To establish the Layer 2.5 VPN, follow the same steps you used to configure PE1. You use MPLS on interfaces pointing toward the core and the edge, establish a Layer 2 VPN BGP peer relationship with PE1, use LDP or RSVP for traffic traveling through the core, and configure the proper VRF instance. Finally, you create policies on PE2 that will set a private community tag on outbound BGP traffic heading to PE1 and accept incoming traffic that matches similar community traffic from PE1.

```

PE2 [edit]
interfaces {
  fe-1/1/2 {
    description "to P fe-1/1/0";
    unit 0 {
      family inet {
        address 192.168.255.94/30;
      }
      family mpls;
    }
  }
  so-1/2/1 {
    description "to CE2 so-1/2/1";
    dce;
    encapsulation frame-relay-tcc;
    unit 600 {
      encapsulation frame-relay-tcc;
      dlci 600;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.255.177/32;
      }
    }
  }
}

```

```

protocols {
  mpls {
    interface fe-1/1/2.0;
    interface so-1/2/1.600;
  }
  bgp {
    group my-internal-peers {
      type internal;
      local-address 10.255.255.177;
      family I2-vpn {
        unicast;
      }
      neighbor 10.255.255.174;
    }
  }
  ldp {
    interface fe-1/1/2.0;
  }
}

policy-options {
  policy-statement companyA-import {
    term T1 {
      from {
        protocol bgp;
        community companyA;
      }
      then accept;
    }
    term Final {
      then reject;
    }
  }
  policy-statement companyA-export {
    term T1 {
      then {
        community add companyA;
        accept;
      }
    }
    term Final {
      then reject;
    }
  }
  community companyA members target:100:1;
}

```

```

routing-instances {
  companyA {
    instance-type l2vpn;
    interface so-1/2/1.600;
    route-distinguisher 10.255.255.177:1;
    vrf-import companyA-import;
    vrf-export companyA-export;
    protocols {
      l2vpn {
        encapsulation-type interworking;
        site NewYork {
          site-identifier 2;
          interface so-1/2/1.600 {
            remote-site-id 1;
          }
        }
      }
    }
  }
}

```

To complete the Layer 2.5 VPN configuration, enable Frame Relay encapsulation on CE2.

```

CE2 [edit]
interfaces
  so-1/2/1 {
    description "to PE2 so-1/2/1";
    encapsulation frame-relay;
    unit 600 {
      dlci 600;
      family inet {
        address 1.1.1.2/30;
      }
    }
  }
}

```

Check Your Work

To verify the operational status of your Layer 2.5 VPN, use the following commands:

```

show route forwarding-table

show ldp database

show l2vpn connections

show bgp summary

show route

```

To view sample output of these commands as used with the configuration example, see the following:

PE1 Status on page 483

PE2 Status on page 485

P Status on page 487

PE1 Status

```
user@PE1> show route forwarding-table
```

```
<snip>
```

```
Routing table:: ccc
```

```
MPLS:
```

Interface.Label	Type	RtRef	Nexthop	Type	Index	NhRef	Netif
default	perm	0		dscd	10	1	
0	user	0		recv	12	2	
1	user	0		recv	12	2	
100128	user	0		Pop			so-1/0/0.0
100128(S=0)	user	0		Pop			so-1/0/0.0
100129	user	0		Swap	100000		so-1/0/0.0
800001	user	0		ucst	137	1	at-0/3/0.32
at-0/3/0. (CCC)	user	0		indr	133	2	
				Push	800000		Push 100000(top)

```
so-1/0/0.0
```

```
<snip>
```

```
user@PE1> show ldp database
```

```
Input label database, 10.255.255.174:0-10.255.255.173:0
```

Label	Prefix
100002	10.255.255.174/32
100000	10.255.255.177/32
3	10.255.255.173/32

```
Output label database, 10.255.255.174:0-10.255.255.173:0
```

Label	Prefix
100128	10.255.255.173/32
100129	10.255.255.177/32
3	10.255.255.174/32

```
user@PE1> show l2vpn connections
```

```
L2VPN Connections:
```

Legend for connection status (St)	Legend for interface status
OR -- out of range	up -- operational
EI -- encapsulation invalid	Dn -- down
EM -- encapsulation mismatch	NP -- no present
CN -- circuit not present	DS -- disabled
OL -- no outgoing label	WE -- wrong encapsulation
Dn -- down	UN -- uninitialized
VC-Dn -- Virtual circuit down	
WE -- intf encaps != instance encaps	
-> -- only outbound conn is up	
<- -- only inbound conn is up	
UP -- operational	
XX -- unknown	

```

Instance: companyA
Local site: Denver (1)
  connection-site      Type St      Time last up      # Up trans
  2                    rmt  Up      Nov 30 08:21:07 2001      1
  Local interface: at-0/3/0.32, Status: Up, Encapsulation: INTERWORKING
  Remote PE: 10.255.255.177
  Incoming label: 800001, Outgoing label: 800000

user@PE1> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed  History  Damp  State  Pending
inet.0         0          0          0           0        0    0      0
bgp.l2vpn.0    1          1          0           0        0    0      0
Peer           AS          InPkt      OutPkt      OutQ     Flaps  Last  Up/Dwn
State|#Active/Received/Damped...
10.255.255.177 69         49         45          0        1    19:16 Establ
  bgp.l2vpn.0: 1/1/0
  companyA.l2vpn.0: 1/1/0

user@PE1> show route

<snip>

mpls.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                *[MPLS/0] 1d 18:54:24, metric 1
  Receive
1                *[MPLS/0] 1d 18:54:24, metric 1
  Receive
100128           *[LDP/9] 00:24:03, metric 1
  > via so-1/0/0.0, Pop
100128(S=0)     *[LDP/9] 00:24:03, metric 1
  > via so-1/0/0.0, Pop
100129           *[LDP/9] 00:24:03, metric 1
  > via so-1/0/0.0, Swap 100000
800001           *[L2VPN/7] 00:10:35
  > via at-0/3/0.32, Pop      [0]

at-0/3/0.32     *[L2VPN/7] 00:10:35
  > via so-1/0/0.0, Push 800000, Push 100000(top)

companyA.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:1:1         /96
  *[L2VPN/7] 00:19:55
  Discard
1:1:2:1         /96
  *[BGP/170] 00:06:46, localpref 100, from 10.255.255.177
  AS path: I
  > via so-1/0/0.0, Push 100000

bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:2:1         /96
  *[BGP/170] 00:10:35, localpref 100, from 10.255.255.177
  AS path: I
  > via so-1/0/0.0, Push 100000

<snip>

```

PE2 Status

```
user@vpn07> show route forwarding-table
```

```
<snip>
```

```
Routing table:: ccc
```

```
MPLS:
```

Interface.Label	Type	RtRef	NextHop	Type	Index	NhRef	Netif
default	perm	0		dscd	8	1	
0	user	0		recv	10	2	
1	user	0		recv	10	2	
100002	user	0		Pop			fe-1/1/2.0
100002(S=0)	user	0		Pop			fe-1/1/2.0
100003	user	0		Swap	100002		fe-1/1/2.0
800000	user	0		ucst	60	1	so-1/2/1.0
so-1/2/1. (CCC)	user	0		indr	59	2	

```
<snip>
```

```
user@vpn07> show ldp database
```

```
Input label database, 10.255.255.177:0-10.255.255.173:0
```

Label	Prefix
100000	10.255.255.177/32
3	10.255.255.173/32
100002	10.255.255.174/32

```
Output label database, 10.255.255.177:0-10.255.255.173:0
```

Label	Prefix
100002	10.255.255.173/32
3	10.255.255.177/32
100003	10.255.255.174/32

```
user@vpn07> show l2vpn connections
```

```
L2VPN Connections:
```

```
Legend for connection status (St) Legend for interface status
```

OR -- out of range	up -- operational
EI -- encapsulation invalid	Dn -- down
EM -- encapsulation mismatch	NP -- no present
CN -- circuit not present	DS -- disabled
OL -- no outgoing label	WE -- wrong encapsulation
Dn -- down	UN -- uninitialized
VC-Dn -- Virtual circuit down	
WE -- intf encaps != instance encaps	
-> -- only outbound conn is up	
<- -- only inbound conn is up	
UP -- operational	
XX -- unknown	

```
Instance: companyA
```

```
Local site: NewYork (2)
```

connection-site	Type	St	Time last up	# Up trans
1	rmt	Up	Nov 30 08:21:01 2001	1

```
Local interface: so-1/2/1.0, Status: Up, Encapsulation: INTERWORKING
```

```
Remote PE: 10.255.255.174
```

```
Incoming label: 800000, Outgoing label: 800001
```

```

user@vpn07> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed  History  Damp State   Pending
bgp.l2vpn.0    1          1          0           0         0     0         0
inet.0         0          0          0           0         0     0         0
Peer          AS        InPkt    OutPkt    OutQ    Flaps Last Up/Dwn
State|#Active/Received/Damped...
10.255.255.174 69        45       52        0        0        20:20 Establ
  bgp.l2vpn.0: 1/1/0
  companyA.l2vpn.0: 1/1/0

user@vpn07> show route

<snip>

mpls.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0          *[MPLS/0] 02:34:04, metric 1
           Receive
1          *[MPLS/0] 02:34:04, metric 1
           Receive
100002     *[LDP/9] 00:25:39, metric 1
           > via fe-1/1/2.0, Pop
100002(S=0) *[LDP/9] 00:25:39, metric 1
           > via fe-1/1/2.0, Pop
100003     *[LDP/9] 00:25:01, metric 1
           > via fe-1/1/2.0, Swap 100002
800000     *[L2VPN/7] 00:07:50
           > via so-1/2/1.0, Pop      [0]
so-1/2/1.0 *[L2VPN/7] 00:07:50
           > via fe-1/1/2.0, Push 800001, Push 100002(top)

companyA.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:1:1    /96
           *[BGP/170] 00:04:59, localpref 100, from 10.255.255.174
           AS path: I
           > via fe-1/1/2.0, Push 100002
1:1:2:1    /96
           *[L2VPN/7] 00:11:34
           Discard

bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:1:1    /96
           *[BGP/170] 00:11:38, localpref 100, from 10.255.255.174
           AS path: I
           > via fe-1/1/2.0, Push 100002

<snip>

```

P Status

```

user@P> show ldp database

Input label database, 10.255.255.173:0-10.255.255.174:0
  Label      Prefix
  100128     10.255.255.173/32
  100129     10.255.255.177/32
  3          10.255.255.174/32

Output label database, 10.255.255.173:0-10.255.255.174:0
  Label      Prefix
  3          10.255.255.173/32
  100000     10.255.255.177/32
  100002     10.255.255.174/32

Input label database, 10.255.255.173:0-10.255.255.177:0
  Label      Prefix
  3          10.255.255.177/32
  100002     10.255.255.173/32
  100003     10.255.255.174/32

Output label database, 10.255.255.173:0-10.255.255.177:0
  Label      Prefix
  3          10.255.255.173/32
  100000     10.255.255.177/32
  100002     10.255.255.174/32

```

For More Information

For additional information on TCC or Layer 2.5 VPNs, see the following documents:

JUNOS Internet Software Configuration Guide: VPNs

JUNOS Internet Software Configuration Guide: MPLS Applications

JUNOS Internet Software Configuration Guide: Network Interfaces and Class of Service

Revision History

30 June 2003—Added T-series support for TCC and Layer 2.5 VPNs, 6.0R1 Release. Elizabeth Lichtenberg.

2 April 2003—5.7R1 Release. Richard Hendricks.

27 December 2002—Added proxy ARP, 5.6R1 Release. Richard Hendricks.

30 September 2002—5.5R1 Release. Richard Hendricks.

19 July 2002—5.4R1 Release. Richard Hendricks.

6 June 2002—Document reformatted, added Ethernet TCC. Richard Hendricks.

21 November 2001—Initial document created. Gary Matthews.

