

## Chapter 19

# Virtual Private LAN Service

Ethernet is an increasingly important component of a service provider's slate of service offerings. Many customers are requesting the ability to connect local area network (LAN) locations across the country and around the world. To fulfill customer desire, service providers have had to set up complex point-to-point Layer 2 virtual private networks (VPNs) or connect expensive Layer 2 switches to handle traffic.

In JUNOS Release 5.7 and later, an emerging service is available to meet the growing Ethernet needs of service providers and their customers. Virtual private LAN service (VPLS) is based on the Internet Engineering Task Force (IETF) Internet draft draft-ietf-l2vpn-vpls-bgp-08.txt, *Virtual Private LAN Service (VPLS) Using BGP for Auto-discovery and Signaling* (expires December 2006). VPLS is an Ethernet-based multipoint-to-multipoint Layer 2 VPN. With VPLS, multiple Ethernet LAN sites can be connected to each other across an MPLS backbone. To the customer, all sites interconnected by VPLS appear to be on the same Ethernet LAN (even though traffic travels across a service provider network).

This guide explains the background knowledge you need to understand VPLS and provides detailed steps for you to follow to implement it in your network.

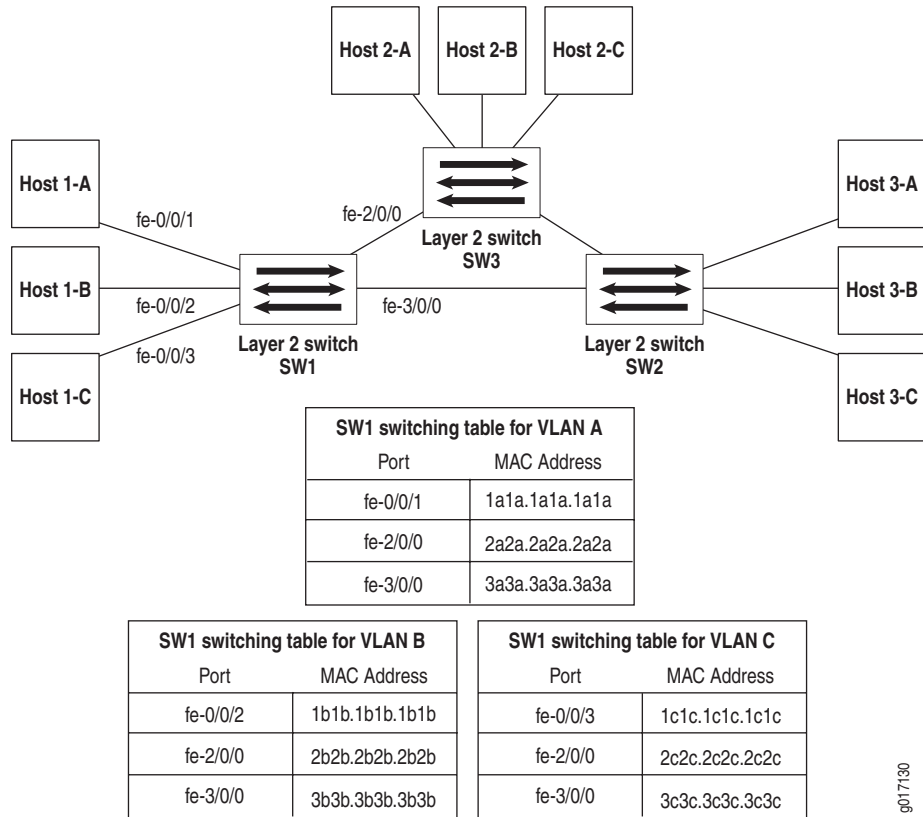
This feature guide covers these topics:

- Overview on page 918
- System Requirements on page 920
- Terms and Acronyms on page 921
- Configuring VPLS on page 922
  - Example: VPLS Configuration on page 937
  - Checking Your Work on page 944
- For More Information on page 949
- Revision History on page 950

## Overview

Before VPLS, the only way you could connect Ethernet LAN sites together was to set up a Layer 2 VPN or install multiple Layer 2 Ethernet switches. Figure 78 shows how three switches can be connected to each other.

**Figure 78: Ethernet Switching Example**

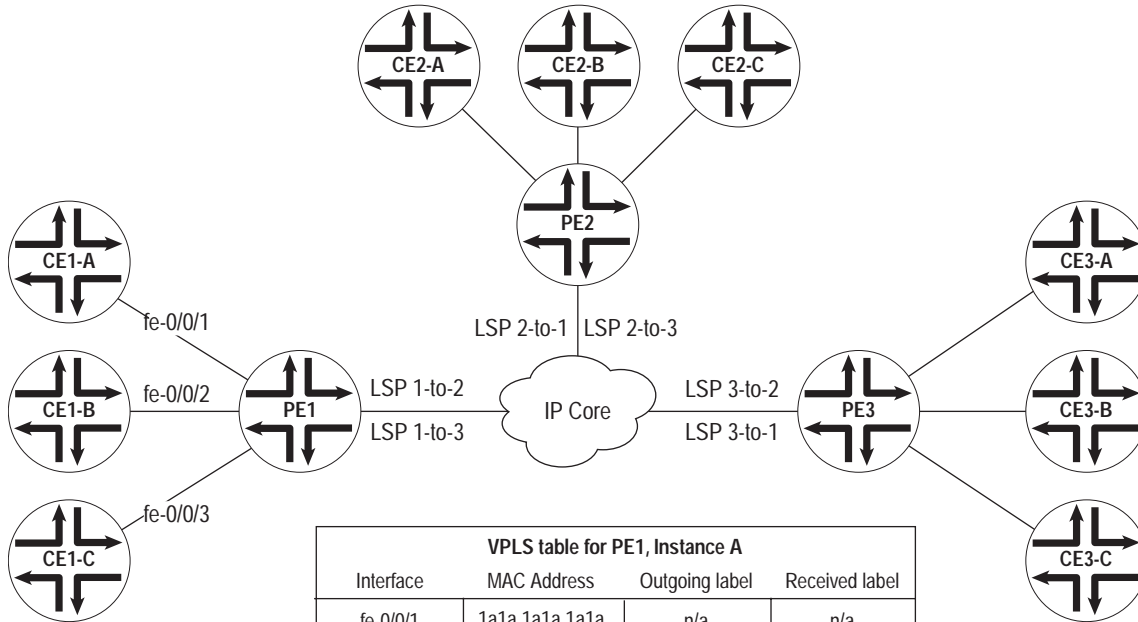


g017130

A typical switch builds its Layer 2 switching table with MAC address and interface information learned from other switches. If a switch does not know how to reach a particular destination, it floods traffic for that destination to all ports except the one where the traffic originated. When a reply for an unknown destination is received, this information is added to the switching table. If a destination is known, the switch sends the traffic directly to the intended recipient through the associated port listed in the switching table.

Figure 79 on page 919 shows a VPLS network comparable to the switch example and explains how VPLS functions similarly to Ethernet switches.

Figure 79: VPLS Introductory Example



VPLS table for PE1, Instance A			
Interface	MAC Address	Outgoing label	Received label
fe-0/0/1	1a1a.1a1a.1a1a	n/a	n/a
vt-0/3/0.32770	2a2a.2a2a.2a2a	800000	800002
vt-0/3/0.32773	3a3a.3a3a.3a3a	800012	800014

VPLS table for PE1, Instance B			
Interface	MAC Address	Outgoing label	Received label
fe-0/0/2	1b1b.1b1b.1b1b	n/a	n/a
vt-0/3/0.32771	2b2b.2b2b.2b2b	800004	800006
vt-0/3/0.32774	3b3b.3b3b.3b3b	800016	800018

VPLS table for PE1, Instance C			
Interface	MAC Address	Outgoing label	Received label
fe-0/0/3	1c1c.1c1c.1c1c	n/a	n/a
vt-0/3/0.32772	2c2c.2c2c.2c2c	800008	800010
vt-0/3/0.32775	3c3c.3c3c.3c3c	800020	800022

g017129

Notice that Layer 2 information gathered by a switch (for example, MAC addresses and interface ports) is included in the VPLS instance table. However, instead of requiring all VPLS interfaces to be physical switch ports, the router allows remote traffic for a VPLS instance to be delivered across an MPLS label-switched path (LSP) and arrive on a virtual port. The virtual port emulates a local, physical port. Traffic can be learned, forwarded, or flooded to the virtual port almost identically to the way traffic is sent to a local port.

The VPLS table learns MAC address and interface information for both physical and virtual ports. If no activity is seen for a particular MAC address, it is purged from the table over time.

As shown in Figure 79, the main difference between a physical port and a virtual port is that the router captures additional information from a virtual port—an outgoing MPLS label used to reach the remote site and an incoming MPLS label for VPLS traffic received from the remote site.

When you configure VPLS on a routing platform, a virtual port is generated as a logical interface on a virtual loopback tunnel (vt) interface or a label-switched interface (LSI). Virtual ports are created dynamically on vt interfaces if you install a Physical Interface Card (PIC) that supports virtual tunnels. For this default version of VPLS, you must install at least one Tunnel Services, Link Services, or Adaptive Services PIC in each VPLS provider edge (PE) router. If your routing platform does not contain a PIC that offers tunnel services, you can configure VPLS to create virtual ports on LSI logical interfaces.

One restriction to flooding behavior in VPLS is that traffic received from remote PE routers is never forwarded to other PE routers. This restriction helps prevent loops in the core network. However, if a customer edge (CE) Ethernet switch has two connections or more to the same PE router, you must enable the Spanning Tree Protocol on the CE switch to prevent loops. (Spanning tree is not supported directly on M-series routers.)



**NOTE:** Juniper Networks routers support transmission of standard Bridge Protocol Data Unit (BPDU) frames across Layer 2 VPNs, Layer 2 circuits, and VPLS instances. However, some CE Ethernet switches that generate proprietary BPDU frames might not be able to run the Spanning Tree Protocol across Juniper Networks routers configured for these emulated Layer 2 connections.

## System Requirements

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

- JUNOS Release 7.6 or later for VPLS support on LSI logical interfaces
- JUNOS Release 7.5 or later for multihoming a CE router to multiple PE routers
- JUNOS Release 7.3 or later for VPLS per-packet load balancing, support for limiting MAC address learning per interface in a VPLS domain, and migration to the VPLS and Layer 2 VPN `signaling` statement at the `[edit protocols bgp groups group-name family l2vpn]` hierarchy level.
- JUNOS Release 6.4 or later to implement Ethernet VPLS over ATM LLC interface encapsulation on T-series and M320 routing platforms, to select the tunnel-enabled PICs that provide virtual ports for VPLS operation, and to issue the `show vpls statistics` command
- JUNOS Release 6.3 or later to clear MAC addresses from the VPLS table and to modify VPLS table timeout intervals
- JUNOS Release 6.2 or later for VPLS class of service (CoS), VPLS graceful restart, VPLS interinstance bridging and routing, VPLS source and destination MAC address accounting, VPLS virtual port support on the Adaptive Services PIC for M-series routers, and general VPLS support for T-series and M320 routing platforms
- JUNOS Release 6.1 or later for VPLS policers and filters
- JUNOS Release 6.0 or later for Ethernet VPLS over ATM LLC interface encapsulation on M-series routers

- JUNOS Release 5.7 or later for Ethernet VPLS, VLAN VPLS, and extended VLAN VPLS interface encapsulations
- Two Juniper Networks M-series (except the M160 router) or T-series routing platforms for the provider edge (PE)
- One Adaptive Services PIC, Link Services PIC, or Tunnel Services PIC per routing platform to create VPLS virtual ports on vt interfaces
- One Fast Ethernet or Gigabit Ethernet PIC per routing platform (from this list):
  - 4-port Fast Ethernet PIC with 10/100 Base-TX interfaces
  - 1-port, 2-port, or 10-port Gigabit Ethernet PIC
  - 4-port, quad-wide Gigabit Ethernet PIC
  - 1- and 2-port Gigabit Ethernet Intelligent Queuing (IQ) PIC
  - 4- and 8-port Gigabit Ethernet IQ2 PIC with small form-factor pluggable transceivers (SFPs)
  - 1-, 2-, and 4-port Gigabit Ethernet PIC with SFPs
  - 1-port 10-Gigabit Ethernet PIC

## Terms and Acronyms

---

- **virtual private LAN service (VPLS)**—An Ethernet-based multipoint-to-multipoint Layer 2 VPN service used for interconnecting multiple Ethernet LANs across an MPLS backbone. VPLS is specified in the IETF draft *Virtual Private LAN Service*. For more information about VPLS, see the *JUNOS VPNs Configuration Guide*.
- **virtual port**—A special logical interface that is generated dynamically when you configure VPLS on a PE router. Virtual ports send and receive VPLS traffic for remote PE routers as if the remote VPLS sites had Ethernet-based interfaces directly connected to the local PE router. To generate virtual ports, VPLS PE routing platforms use logical interfaces on a vt interface (that is generated by the Tunnel Services PIC, Link Services PIC, or Adaptive Services PIC) or an LSI interface.

## Configuring VPLS

---

To implement VPLS, you must configure the following:

- Configuring BGP, MPLS, RSVP, and an IGP on the PE and Core Routers on page 923
- Configuring VPLS Encapsulation on CE-Facing Interfaces on page 923
- Configuring a VPLS Routing Instance on page 925
- Option: Configuring VPLS to Use LSI Interfaces on page 926
- Option: Selecting an LSP for the VPLS Routing Instance to Traverse on page 927
- Option: Applying VPLS Policers and Filters on page 928
- Option: Enabling VPLS Class of Service on page 931
- Option: Enabling VPLS Graceful Restart on page 931
- Option: Clearing MAC Addresses and Modifying the VPLS Table Timeout Interval on page 931
- Option: Configuring VPLS Interinstance Bridging and Routing on page 932
- Option: Selecting PICs to Process VPLS Traffic on page 933
- Option: Limiting the Number of MAC Addresses Learned on an Interface on page 934
- Option: Optimizing VPLS Traffic Flows on page 934
- Option: VPLS Multihoming on page 935

To apply your knowledge, visit these sections:

- Example: VPLS Configuration on page 937
- Checking Your Work on page 944

## Configuring BGP, MPLS, RSVP, and an IGP on the PE and Core Routers

At a fundamental level, VPLS is a type of Layer 2 VPN. All forms of Layer 2 VPNs require you to configure network protocols to handle *intradomain routing* (an interior gateway protocol [IGP], such as Open Shortest Path First [OSPF] or Intermediate System-to-Intermediate System [IS-IS]), *interdomain routing* (Border Gateway Protocol [BGP]), *label switching* (Multiprotocol Label Switching [MPLS]), and *path signaling* (Resource Reservation Protocol [RSVP]). For more information about these protocols and examples of how to configure these protocols to support a Layer 2 VPN, see the *JUNOS VPNs Configuration Guide*.



**NOTE:** The 8-port, 12-port, and 48-port dense Fast Ethernet Physical Interface Cards (PICs) cannot push more than two labels onto an MPLS packet. Because of this, we do not recommend that you configure these PICs as core-facing or equivalent interfaces.

## Configuring VPLS Encapsulation on CE-Facing Interfaces

There are four types of VPLS interface encapsulation: Ethernet VPLS, Ethernet VPLS over ATM LLC, VLAN VPLS, and extended VLAN VPLS. When one of these encapsulations is applied to an interface, a family type of VPLS is enabled by default. The encapsulation types are:

- **ether-vpls-over-atm-llc**—Use Ethernet VPLS over ATM LLC encapsulation on ATM2 IQ logical interfaces. This encapsulation type enables a VPLS instance to support bridging between Ethernet interfaces and ATM interfaces, as described in RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*. When you use this encapsulation type, you configure it on logical interfaces only and you cannot configure multipoint interfaces.
- **extended-vlan-vpls**—Use extended VLAN VPLS encapsulation on Ethernet interfaces that have VLAN 802.1Q tagging and VPLS enabled and that must accept packets carrying TPIDs 0x8100, 0x9100, and 0x9901.



**NOTE:** The built-in Gigabit Ethernet PIC on the M7i router does not support extended VLAN VPLS encapsulation.

- **ethernet-vpls**—Use Ethernet VPLS encapsulation on Ethernet interfaces that have VPLS enabled and must accept packets carrying standard Tag Protocol ID (TPID) values.
- **vlan-vpls**—Use VLAN VPLS encapsulation on Ethernet interfaces with VLAN tagging enabled. VLAN VPLS encapsulation supports TPID 0x8100 only. You must configure this encapsulation type on both the physical interface and the logical interface.

Use the following guidelines to configure a VPLS interface:

- For encapsulation type `vlan-vpls`, VLAN IDs 1 through 511 are reserved for normal Ethernet VLANs, IDs 512 through 1023 are reserved for VPLS VLANs on Fast Ethernet interfaces, and IDs 512 through 4094 are reserved for VPLS VLANs on Gigabit Ethernet interfaces. For encapsulation type `extended-vlan-vpls`, all VLAN IDs from 1 through 1023 are valid for VPLS VLANs on Fast Ethernet interfaces, and all VLAN IDs from 1 through 4094 are valid for VPLS VLANs on Gigabit Ethernet interfaces. VLAN ID 0 is reserved for priority tagging.
- For VLAN-based VPLS, you can configure only one VLAN ID per VPLS instance.

To configure VPLS interface encapsulation for an Ethernet interface, include the `encapsulation` statement at the `[edit interfaces interface-fpc/pic/port]` hierarchy level and select `ethernet-vpls`, `vlan-vpls`, or `extended-vlan-vpls` as the encapsulation type. If you select the VLAN VPLS encapsulation, also include the `vlan-vpls` statement at the `[edit interfaces ethernet-interface-fpc/pic/port unit unit-number encapsulation]` logical interface hierarchy level. When using either VLAN VPLS or extended VLAN VPLS encapsulations, include the `vlan-tagging` statement at the `[edit interfaces ethernet-interface-fpc/pic/port]` hierarchy level.

To configure VPLS interface encapsulation for an ATM2 IQ interface, include the `encapsulation` statement at the `[edit interfaces at-fpc/pic/port]` hierarchy level and select `ether-vpls-over-atm-llc` as the encapsulation type.

```
[edit]
interfaces {
  fe-0/1/0 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 0 {
      encapsulation vlan-vpls;
      vlan-id 600;
    }
  }
  at-0/2/0 {
    encapsulation ether-vpls-over-atm-llc;
  }
}
```



**NOTE:** On a provider edge (PE) router running VPLS, you must configure `ethernet-vpls` encapsulation on a customer-facing interface when you configure the Multiple Spanning Tree Protocol (MSTP). VLAN-based VPLS interface encapsulations are not supported with MSTP.

---

## Configuring a VPLS Routing Instance

Like other Layer 2 VPNs, you must enable a routing instance to isolate VPLS traffic from other network traffic. To configure, include the `instance-type vpls` statement at the `[edit routing-instances instance-name]` hierarchy level.

Within the instance, you can define the maximum number of sites that can participate in this VPLS instance, the size of the MAC address table, a local site name, and a local site identifier. To configure the maximum number of sites, include the `site-range` statement at the `[edit routing-instances instance-name protocols vpls]` hierarchy level. To configure the size of the MAC address table, include the `mac-table-size` statement at the `[edit routing-instances instance-name protocols vpls]` hierarchy level. The default size is 512 addresses, the minimum is 16 addresses, and the maximum is 65,536 addresses.

To configure a site name, include the `site` statement at the `[edit routing-instances instance-name protocols vpls]` hierarchy level. To configure the site ID, include the `site-identifier` statement at the `[edit routing-instances instance-name protocols vpls site name]` hierarchy level.

```
[edit routing-instances]
instance-name {
  instance-type vpls;
  interface fe-0/1/0.0;
  route-distinguisher 10.245.14.218:1;
  vrf-target target:11111:1;
  protocols {
    vpls {
      site-range 10;
      mac-table-size 1024;
      site greenPE1 {
        site-identifier 1;
        interface fe-0/1/0.0;
      }
    }
  }
}
```



**NOTE:** If you configure logical routers in your routing platform, you can configure VPLS only in the main router at the `[edit routing-instances instance-name protocols vpls]` hierarchy level.

### Option: Configuring VPLS to Use LSI Interfaces

By default, VPLS uses tunnel-based PICs to create virtual ports on `vt` interfaces. If you do not install a tunnel-based PIC in your routing platform, you can still configure VPLS by using LSI interfaces to support the virtual ports. Use of LSI interfaces requires the use of Ethernet-based PICs installed in an Enhanced FPC.

To use LSI interfaces for VPLS instead of `vt` interfaces, include the `no-tunnel-services` statement at the `[edit routing-instances instance-name protocols vpls]` hierarchy level.

```
[edit routing-instances]
instance-name {
  protocols {
    vpls {
      no-tunnel-services;
    }
  }
}
```



**NOTE:** The following interface types do not support the use of LSI interfaces with VPLS:

- Aggregated SONET/SDH (cannot be used as the core-facing interface)
  - 1-port Gigabit Ethernet
  - 2-port Gigabit Ethernet
  - 4-port Fast Ethernet
-

### Option: Selecting an LSP for the VPLS Routing Instance to Traverse

If you have two or more equal-cost-path LSPs between your VPLS PE router sites, you can select an LSP over which the VPLS traffic will travel. You can assign the VPLS routing instance to a BGP community, define a policy that directs community traffic over a specified LSP, and then apply the policy to the forwarding table.

To configure a BGP community, include the `community community-name` statement at the [edit policy-options] hierarchy level. Be sure to specify the `vrf-export` or `vrf-target` values from the VPLS routing instance as community identifiers with the `members community-ids` statement at the [edit policy-options community community-name] hierarchy level.

To create a policy that sends community traffic over a specific LSP, include the `community community-name` statement at the [edit policy-options policy-statement policy-name term term-name from] hierarchy level and the `install-nexthop lsp lsp-name` statement at the [edit policy-options policy-statement policy-name term term-name then] hierarchy level. To apply the policy to the forwarding table, include the `export policy-name` statement at the [edit routing-options forwarding-table] hierarchy level.

```
[edit]
routing-options {
  autonomous-system 69;
  forwarding-table {
    export LSP-policy;
  }
  policy-options {
    policy-statement LSP-policy {
      term a {
        from community gold;
        then {
          install-nexthop lsp pe1-to-pe2;
          accept;
        }
      }
    }
  }
  community gold members target:11111:1;
}
```

### Option: Applying VPLS Policers and Filters

You can use filters, policers, and broadcast/unknown filters to determine which MAC addresses will be allowed into or out of a VPLS domain. You can apply these filters and policers to CE-facing interfaces only.

**VPLS Policers** To process traffic as it enters a VPLS domain, you can define a firewall policer and apply it to the input interface. To define policer characteristics for incoming VPLS traffic, include the `bandwidth-limit` and `burst-size-limit` statements at the `[edit firewall policer policer-name if-exceeding]` hierarchy level. Then, specify statements to implement the desired action (for example, `discard`) for the policed traffic at the `[edit firewall policer policer-name then]` hierarchy level. To apply the policer to a CE-facing interface, include the `input` or `output` statements and the name of the policer at the `[edit interfaces interface-name unit unit-number family vpls policer]` hierarchy level.

```
[edit]
interfaces {
  fe-2/1/0 {
    vlan-tagging;
    mtu 1544;
    encapsulation vlan-vpls;
    unit 0 {
      encapsulation vlan-vpls;
      vlan-id 600;
      family vpls {
        policer {
          input vpls-policer;
        }
      }
    }
  }
}
firewall {
  policer {
    vpls-policer {
      if-exceeding {
        bandwidth-limit 5m;
        burst-size-limit 1m;
      }
      then discard;
    }
  }
}
```

**VPLS Filters** To process traffic as it exits a VPLS domain, you can define a firewall filter and apply it to the output interface. To configure match conditions for a firewall filter, include the `interface-group`, `source-mac-address`, `destination-mac-address`, `ethernet-type`, or `vlan-ethernet-type` statements at the `[edit firewall family vpls filter filter-name term term-name from]` hierarchy level. Then, implement the desired action (for example, `discard`) for the traffic at the `[edit firewall family vpls filter filter-name term term-name then]` hierarchy level. To apply the filter to a CE-facing interface, include the `input`, `output`, or `group` statements at the `[edit interfaces interface-name unit unit-number family vpls filter]` hierarchy level.

```
[edit]
interfaces {
  fe-2/1/1 {
    vlan-tagging;
    mtu 1544;
    encapsulation vlan-vpls;
    unit 0 {
      encapsulation vlan-vpls;
      vlan-id 600;
      family vpls {
        filter {
          output vpls-out-filter;
        }
      }
    }
  }
}
firewall {
  family vpls {
    filter vpls-out-filter {
      interface-specific;
      term 1 {
        from {
          source-mac-address {
            00.10.10.10.11.18/48;
          }
        }
        then {
          count count.ce2;
          accept;
        }
      }
      term 2 {
        then accept;
      }
    }
  }
}
```



**NOTE:** Output filters do not work for broadcast and multicast traffic, including VPLS traffic destined for a multicast MAC address.

**VPLS Broadcast and Unknown Filters**

To restrict the flow of broadcast and unknown packets into a VPLS domain, you must create a firewall filter and apply the filter to one of the forwarding tables of the VPLS routing instance. When you apply a filter in this way, the filter processes traffic from all interfaces in the instance, including vt interfaces. To configure match conditions for a VPLS-based firewall filter, include the `source-mac-address`, `destination-mac-address`, `interface-group`, `ethernet-type`, or `vlan-ethernet-type` statements at the `[edit firewall family vpls filter filter-name term term-name from]` hierarchy level. Then, specify statements to activate the desired action (for example, `discard`) for the matched packets at the `[edit firewall family vpls filter filter-name term term-name then]` hierarchy level.

To apply the filter to the broadcast and unknown table of a VPLS routing instance, include the `input` statement and the name of the filter at the `[edit routing-instances instance-name forwarding-options family vpls flood]` hierarchy level. To apply the filter to the destination MAC address table of a VPLS routing instance, include the `input` statement and the name of the filter at the `[edit routing-instances instance-name forwarding-options family vpls filter]` hierarchy level.

```
[edit]
firewall {
  family vpls {
    filter vpls-flood {
      term 1 {
        from {
          destination-mac-address {
            00.90.69.dc.95.3b/48;
          }
        }
        then discard;
      }
      term 2 {
        then accept;
      }
    }
  }
}
routing-instances {
  green {
    forwarding-options {
      family vpls {
        (flood | filter) {
          input vpls-flood;
        }
      }
    }
  }
}
```

When you configure VPLS, a priority filter for Spanning Tree Protocol (STP) bridge protocol data units (BPDUs) is enabled by default. This BPDU filter matches on the well-known STP MAC address of `01:80:c2:00:00:00/24` and applies high priority to this traffic.

For more information on VPLS policers and filters, see the *JUNOS Policy Framework Configuration Guide* and the *JUNOS VPNs Configuration Guide*.

**Option: Enabling VPLS Class of Service**

For JUNOS Release 6.2 or later, you can configure class of service (CoS) for all interfaces in the VPLS domain. CoS information is sent across the MPLS backbone and is preserved for all VPLS traffic processed by local interfaces, virtual ports, and remote interfaces.

For more information on configuring CoS, see the *JUNOS Class of Service Configuration Guide*.

**Option: Enabling VPLS Graceful Restart**

VPLS graceful restart allows you to continue forwarding VPLS traffic across the core MPLS network even if one of the routers in the forwarding path restarts. Graceful restart for VPLS functions the same way as Layer 2 VPN graceful restart. To configure graceful restart for VPLS, include the `graceful-restart` statement at the [edit routing-options] hierarchy level on all PE and core routers.

```
[edit]
routing-options {
    graceful-restart;
}
```

For more information on graceful restart, see “Graceful Restart” on page 413.

**Option: Clearing MAC Addresses and Modifying the VPLS Table Timeout Interval**

You can fine-tune your VPLS domain by clearing MAC address entries from the VPLS table or modifying the default timeout interval for the VPLS table.

To clear all MAC address entries from the VPLS table, issue the `clear vpls mac-address` command. Add the `logical-router logical-router-name` option to clear entries within a logical router and include the `instance instance-name` option to clear entries in a specific VPLS instance. Use the `mac-address` option to remove individual MAC addresses.

To configure the VPLS table timeout interval, include the `mac-table-aging-time` statement at the [edit routing-instances instance-name protocols vpls] hierarchy level. The default interval is 300 seconds, with a minimum of 10 seconds and a maximum of 1 million seconds. As a general rule, you can configure longer values for small, stable VPLS networks and shorter values for large, dynamic VPLS networks. If the VPLS table does not receive any updates during the timeout interval, the router waits one additional interval before automatically clearing MAC address entries from the VPLS table.

```
[edit]
routing-instances {
    instance-name {
        protocols {
            vpls {
                mac-table-aging-time seconds;
            }
        }
    }
}
```

### Option: Configuring VPLS Interinstance Bridging and Routing

To deliver interinstance traffic between two or more VPLS instances, or between a VPLS instance and a Layer 3 VPN routing instance, you must use a logical tunnel interface. Originally designed to interconnect logical routers, the logical tunnel interface acts as a point-to-point connection between instances. A logical tunnel interface can be generated by a Tunnel Services PIC installed on an Enhanced FPC in your routing platform or an integrated Adaptive Services Module installed in an M7i router. To configure a logical tunnel interface, include the `lt-fpc/pic/O` statement at the `[edit interfaces]` hierarchy level. Keep in mind these rules when you connect instances:

- You need to configure both endpoints of the logical tunnel. Configure the first logical tunnel interface in the VPLS instance and the second within the instance you want to interconnect to the VPLS domain.
- Choose one of several interface encapsulation types for your logical tunnel interface peers. Your choices are Ethernet, Ethernet circuit cross-connect (CCC), Ethernet VPLS, Frame Relay, Frame Relay CCC, VLAN, VLAN CCC, and VLAN VPLS. Include one of these choices with the `encapsulation` statement at the `[edit interfaces lt-fpc/pic/O unit unit-number]` hierarchy level.
- Depending on the encapsulation type you select, specify a corresponding data-link connection identifier (DLCI) number for Frame Relay or a VLAN identifier for VLAN encapsulations on your logical tunnel interface peers. To configure, include the `dlci` or `vlan-id` statement at the `[edit interfaces lt-fpc/pic/O unit unit-number]` hierarchy level.
- Your choice of protocol family for the logical tunnel interface also is determined by your selection of an encapsulation type. For Ethernet VPLS and VLAN VPLS, family `vpls` is assigned by default. For all other Ethernet and VLAN encapsulation types, include the `mpls` or `inet` statement at the `[edit interfaces lt-fpc/pic/O unit unit-number family]` hierarchy level. For Frame Relay encapsulation types, you can configure any of the available protocol families: `ccc`, `inet`, `inet6`, `iso`, `mpls`, or `tcc`.
- Be sure to match the logical interface unit numbers of the peering logical tunnel interfaces. To configure, include the `peer-unit` statement at the `[edit interfaces lt-fpc/pic/O unit unit-number]` hierarchy level.

```
[edit]
interfaces {
  lt-fpc/pic/O {
    unit unit-number {
      encapsulation (ethernet | ethernet-ccc | ethernet-vpls | frame-relay |
                    frame-relay-ccc | vlan | vlan-ccc | vlan-vpls);
      peer-unit number; # The logical unit number of the peering lt interface.
      dlci dci-number;
      vlan-id vlan-number;
      family (ccc | inet | inet6 | iso | mpls | tcc);
    }
  }
}
```

```

routing-instances {
  vpls-instance-name {
    interface ge-fpc/pic/port.unit-number;
    interface lt-0/0/0.1;
    ...
  }
  second-instance-name {
    interface at-fpc/pic/port.unit-number;
    interface lt-0/0/0.2;
    ...
  }
}

```

### Option: Selecting PICs to Process VPLS Traffic

The PICs that can create VPLS virtual ports dynamically from vt interfaces include the Tunnel Services PIC, the Link Services PIC, and the Adaptive Services PIC. By default, the JUNOS software automatically and randomly selects vt interfaces to act as VPLS virtual ports in a round-robin fashion. However, if your routing platform contains two or more of these tunnel-enabled PICs, you can manually select which PICs process traffic for each VPLS domain.

You can select a PIC to be the primary device responsible for VPLS traffic processing. You can also select a group of PICs to share responsibility for VPLS traffic processing. When the primary PIC is operating normally, it handles all VPLS-related tasks. If the primary device is not available, any PICs included in the VPLS PIC group assume responsibility.

To select a PIC to become the primary device responsible for VPLS traffic processing, include the `primary` statement at the `[edit routing-instances instance-name protocols vpls tunnel-services]` hierarchy level. To select a group of PICs to share responsibility for VPLS traffic processing, include the `devices` statement at the `[edit routing-instances instance-name protocols vpls tunnel-services]` hierarchy level.

```

[edit]
routing-instances {
  instance-name {
    protocols {
      vpls {
        tunnel-services {
          devices [vt-0/0/0 vt-1/0/0 vt-2/0/0];
          primary vt-0/0/0;
        }
      }
    }
  }
}

```

### Option: Limiting the Number of MAC Addresses Learned on an Interface

You can limit the number of MAC addresses learned for an entire VPLS domain, or from a specific interface within the domain. To set limits for all interfaces assigned to a VPLS domain, include the `interface-mac-limit` statement at the `[edit routing-instances instance-name protocols vpls]` hierarchy level. To set limits for a specific interface within a VPLS domain, include the `interface-mac-limit` statement at the `[edit routing-instances instance-name protocols vpls site site-name interface interface-name]` hierarchy level.

The range of values for the `interface-mac-limit` statement is 16 through 65536. The output of the `show vpls statistics` command displays the results of configuring interface-level MAC address limitations.

```
[edit]
routing-instances {
  instance-name {
    protocols {
      vpls {
        interface-mac-limit number;
        site site-name {
          interface interface-name {
            interface-mac-limit number;
          }
        }
      }
    }
  }
}
```

### Option: Optimizing VPLS Traffic Flows

To improve the performance of VPLS traffic processing in your routing platform, you can implement the following features:

- To optimize VPLS traffic flows across multiple paths, you can enable per-packet load balancing. To configure, include the `load-balance per-packet` statement at the `[edit policy-options policy-statement policy-name term term-name then]` hierarchy level and apply the policy to the forwarding table with the `export policy-name` statement at the `[edit routing-options forwarding-table]` hierarchy level.
- To optimize hashing of source and destination MAC addresses within VPLS traffic flows, include the `source-mac` and `destination-mac` statements at the `[edit forwarding-options hash-key family multiservice]` hierarchy level.

For more information on load balancing and hash keys, see the *JUNOS Policy Framework Configuration Guide*.

### Option: VPLS Multihoming

You can connect a CE device (such as a router or switch) to multiple PE routers in a VPLS network to provide redundancy. A VPLS site multihomed to two or more PE routers provides redundant connectivity if a PE router-to-CE device link fails or a PE router fails.

To allow a VPLS site to be multihomed, you must assign the same site ID on all PE routers connected to the same CE devices, configure the same route distinguisher, reference all interfaces assigned to the site on each PE router, specify a primary interface, and configure multihoming for the site.

To assign a site ID, include the `site-identifier` statement at the `[edit routing-instances instance-name protocols vpls site name]` hierarchy level. To configure a route distinguisher, include the `route-distinguisher` statement at the `[edit routing-instances instance-name]` hierarchy level. To specify the interfaces associated with a site, include the `interface` statement at the `[edit routing-instances instance-name protocols vpls site name]` hierarchy level.

To specify a multihomed interface as the primary interface for a site, include the `active-interface` statement at the `[edit routing-instances instance-name protocols vpls site name]` hierarchy level. If there are multiple interfaces, the remaining interfaces are activated only when the primary interface goes down. You can configure a specific primary interface or use the `any` option. If no active interfaces are configured at the site level, it is assumed that all traffic for a VPLS site travels through a single, nonmultihomed PE router.

When a CE device is connected to the same VPLS site on more than one PE router, include the `multi-homing` statement on all associated PE routers at the `[edit routing-instances instance-name protocols vpls site name]` hierarchy level. Configuration of this statement tracks BGP peers. If no BGP peer is available, VPLS deactivates all active interfaces for a site.



**NOTE:** If you add a direct connection between CE devices that are multihomed to the same VPLS site on different PE routers, traffic loops and loss of connectivity might occur. We do not recommend this topology.

---

The following example shows how a single CE device is multihomed to two PE devices by configuring the same site name, site ID, and route distinguisher at the [edit routing-instances] hierarchy level:

```

Router 1 [edit]
            routing-instances {
            green {
            instance-type vpls;
            interface fe-0/1/3.0;
            route-distinguisher 10.255.14.218:1;
            vrf-target target:11111:1;
            protocols {
            vpls {
            site-range 10;
            site green4{
            site-identifier 4;
            multi-homing;
            active-interface any;
            interface fe-1/1/3.0;
            }
            }
            }
            }
            }

```

```

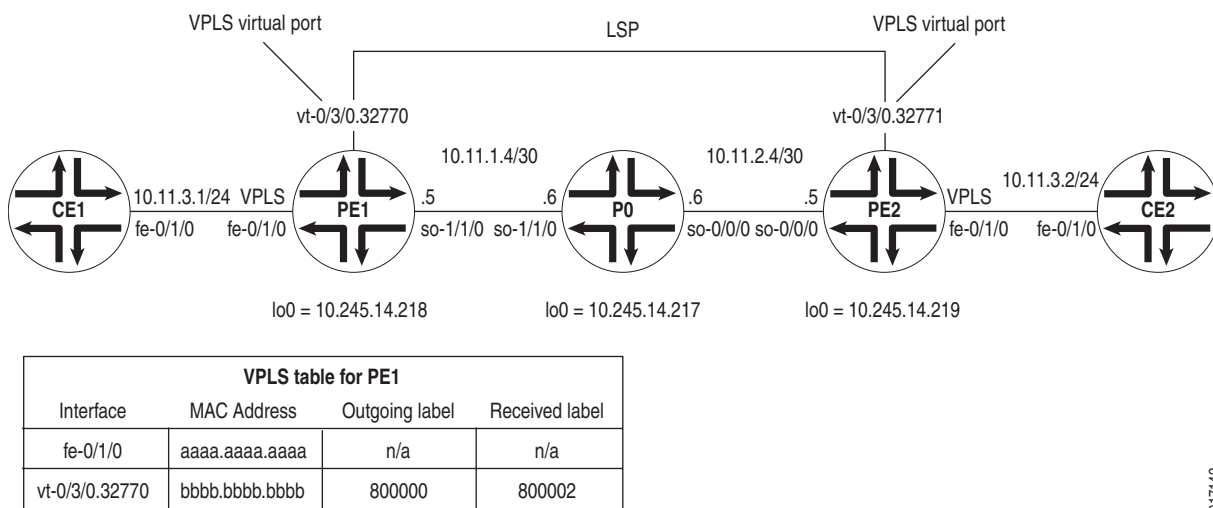
Router 2 [edit]
            routing-instances {
            green {
            instance-type vpls;
            interface fe-0/1/0.0;
            route-distinguisher 10.255.14.218:1;
            vrf-target target:11111:1;
            protocols {
            vpls {
            site-range 10;
            site green4{
            site-identifier 4;
            multi-homing;
            active-interface any;
            interface fe-0/1/0.0;
            }
            }
            }
            }
            }

```

For more information on VPLS multihoming, see the *JUNOS VPNs Configuration Guide*.

## Example: VPLS Configuration

Figure 80: VPLS Topology Diagram



In Figure 80, a simple VPLS topology is enabled between routers PE1 and PE2. CE routers CE1 and CE2 use Ethernet-based interfaces to connect VLAN 600 to their local PE router. The PE routers PE1 and PE2 are connected to one another by LSPs enabled across a service provider backbone running MPLS, BGP, RSVP, and OSPF.

In a VPLS routing instance named **green**, PE1 has a local interface **fe-0/1/0** and a virtual port of **vt-0/3/0.32770** (the virtual port is created dynamically on the Tunnel Services PIC when VPLS is configured). PE2 has a local interface **fe-0/1/0** and a virtual port of **vt-0/3/0.32771** in the same **green** instance. As a result, routers CE1 and CE2 can send Ethernet traffic to one another as if they are physically connected to each other on a LAN.

On Router CE1, the only item you need to configure is the Fast Ethernet interface that connects to PE1. Be sure to write down the VLAN identifier and IP address, so you can match them later on CE2.

```

Router CE1 [edit]
interfaces {
  fe-0/1/0 {
    vlan-tagging; # Configure VLAN tagging for VLAN VPLS or extended VLAN VPLS.
    unit 0 {
      vlan-id 600; # The Ethernet interface on CE2 must use the same VLAN ID.
      family inet {
        address 10.11.3.1/24; # The interface on CE2 must use the same prefix.
      }
    }
  }
}

```

On Router PE1, prepare the router for VPLS by configuring BGP, MPLS, OSPF, and RSVP. (These protocols are the basis for most Layer 2 VPN-related applications, including VPLS.) Include the **signaling** statement at the [edit protocols bgp group *group-name* family l2vpn] hierarchy level, because VPLS uses the same infrastructure for internal BGP as Layer 2 VPNs.



**NOTE:** In JUNOS Release 7.3 and later, the **signaling** statement replaces the **unicast** statement at the [edit protocols bgp group *group-name* family l2vpn] hierarchy level. You must use the **signaling** statement if you wish to configure VPLS domains and Layer 2 VPNs simultaneously.

Next, configure VLAN tagging on the Fast Ethernet interface connected to Router CE1. Include VLAN VPLS encapsulation at both the physical and logical interface levels. Be sure to use the same VLAN ID for all Ethernet interfaces that are part of a single VPLS instance. Finally, add the Fast Ethernet interface into a VPLS routing instance and specify the site range, site ID number, and site name.

```

Router PE1 [edit]
interfaces {
  fe-0/1/0 {
    vlan-tagging; # Configure VLAN tagging for VLAN VPLS or extended VLAN VPLS.
    encapsulation vlan-vpls; # Configure VPLS encapsulation on both the
    unit 0 { # physical interface and the logical interface.
      encapsulation vlan-vpls;
      vlan-id 600; # The VLAN ID is the same one used by the CE routers.
    } # No IP address is needed on the CE-facing interface.
  }
  so-1/1/0 {
    unit 0 {
      family inet {
        address 10.11.1.5/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.245.14.218/32;
      }
    }
  }
}
routing-options {
  autonomous-system 69;
  forwarding-table {
    export exp-to-fwd; # Applies a policy that selects an LSP for the VPLS instance.
  }
}

```

```

protocols {
  rsvp {
    interface all {
      aggregate;
    }
  }
  mpls {
    label-switched-path pe1-to-pe2 { # Configure an LSP to reach other VPLS PEs.
      to 10.245.14.219;
    }
    interface all;
  }
  bgp {
    group vpls-pe {
      type internal;
      local-address 10.245.14.218;
      family l2vpn { # VPLS uses the same infrastructure as Layer 2 VPNs
        signaling; # for internal BGP.
      }
      neighbor 10.245.14.217;
      neighbor 10.245.14.219;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-1/1/0.0 {
        metric 11;
      }
      interface lo0.0 {
        passive;
      }
    }
  }
}
policy-options {
  policy-statement exp-to-fwd {
    term a {
      from community grn-com; # Matches the community in the VPLS instance.
      then {
        install-nexthop lsp pe1-to-pe2; # If there are multiple LSPs that exist
        accept; # between VPLS PE routers, this statement sends VPLS traffic
      } # over a specific LSP.
    }
  }
  community grn-com members target:11111:1; # Adds the instance to a
  # BGP community.
}

```

```

routing-instances {
  green {
    instance-type vpls;      # Configure a VPLS routing instance.
    interface fe-0/1/0.0;
    route-distinguisher 10.245.14.218:1;
    vrf-target target:11111:1; # This value is important to the BGP community.
    protocols {
      vpls {                # Configure a VPLS site range, site name, and site identifier.
        site-range 10;
        site greenPE1 {
          site-identifier 1;
        }
      }
    }
  }
}

```

On Router P0, configure BGP, MPLS, OSPF, and RSVP to interconnect PE1 and PE2.

```

Router P0 [edit]
interfaces {
  so-0/0/0 {
    unit 0 {
      family inet {
        address 10.11.2.6/30;
      }
      family mpls;
    }
  }
  so-1/1/0 {
    unit 0 {
      family inet {
        address 10.11.1.6/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.245.14.217/32;
      }
    }
  }
}

```

```

protocols {
  rsvp {
    interface all {
      aggregate;
    }
  }
  mpls {
    interface all;
  }
  bgp {
    group vpls-pe {
      type internal;
      local-address 10.245.14.217;
      family l2vpn { # VPLS uses the same infrastructure as Layer 2 VPNs
        signaling; # for internal BGP.
      }
      neighbor 10.245.14.218;
      neighbor 10.245.14.219;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-1/1/0.0 {
        metric 11;
      }
      interface so-0/0/0.0 {
        metric 15;
      }
      interface lo0.0 {
        passive;
      }
    }
  }
}

```

On Router PE2, configure BGP, MPLS, OSPF, and RSVP to complement the configuration on PE1. Next, configure VLAN tagging on the Fast Ethernet interface connected to Router CE2. Include VLAN VPLS encapsulation at both the physical and logical interface levels. Be sure to use the same VLAN ID for all Ethernet interfaces that are part of a single VPLS instance. Finally, add the Fast Ethernet interface into a VPLS routing instance and specify the site range, site ID number, and site name.

```

Router PE2 [edit]
interfaces {
  fe-0/1/0 {
    vlan-tagging; # Configure VLAN tagging for VLAN VPLS or extended VLAN VPLS.
    encapsulation vlan-vpls; # Configure VPLS encapsulation on both the
    unit 0 { # physical interface and logical interface.
      encapsulation vlan-vpls;
      vlan-id 600; # The VLAN ID is the same one used by the CE routers.
    } # No IP address is needed on the CE-facing interface.
  }
  so-0/0/0 {
    unit 0 {
      family inet {
        address 10.11.2.5/30;
      }
      family mpls;
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.245.14.219/32;
      }
    }
  }
}
routing-options {
  autonomous-system 69;
  forwarding-table {
    export exp-to-fwd; # Applies a policy that selects an LSP for the VPLS instance.
  }
}

```

```

protocols {
  rsvp {
    interface all {
      aggregate;
    }
  }
  mpls {
    label-switched-path pe2-to-pe1 { # Configure an LSP to other VPLS PE routers.
      to 10.245.14.218;
    }
    interface all;
  }
  bgp {
    group vpls-pe {
      type internal;
      local-address 10.245.14.219;
      family l2vpn { # VPLS uses the same infrastructure as Layer 2 VPNs
        signaling; # for internal BGP.
      }
      neighbor 10.245.14.217;
      neighbor 10.245.14.218;
    }
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface so-0/0/0.0 {
        metric 15;
      }
      interface lo0.0 {
        passive;
      }
    }
  }
}
policy-options {
  policy-statement exp-to-fwd {
    term a {
      from community grn-com; # Matches the community with the VPLS instance.
      then {
        install-nexthop lsp pe2-to-pe1; # If there are multiple LSPs that exist
        accept; # between VPLS PE routers, this statement sends VPLS traffic
      } # over a specific LSP.
    }
  }
  community grn-com members target:11111:1; # This adds the instance into a
  # BGP community.
}

```

```

routing-instances {
  green {
    instance-type vpls;      # Configure a VPLS routing instance.
    interface fe-0/1/0.0;
    route-distinguisher 10.245.14.219:1;
    vrf-target target:11111:1; # This value is important for the BGP community.
    protocols {
      vpls {                # Configure a VPLS site range, site name, and site identifier.
        site-range 10;
        site greenPE2 {
          site-identifier 2;
        }
      }
    }
  }
}

```

On Router CE2, complete your VPLS network by configuring the Fast Ethernet interface that connects to PE2. Use the same VLAN identifier and IP address prefix used on Router CE1.

```

Router CE2 [edit]
interfaces {
  fe-0/1/0 {
    vlan-tagging; # Configure VLAN tagging for VLAN VPLS or extended VLAN VPLS.
    unit 0 {
      vlan-id 600; # The Ethernet interface on CE1 must use the same VLAN ID.
      family inet {
        address 10.11.3.2/24; # The interface on CE1 must use the same prefix.
      }
    }
  }
}

```

## Checking Your Work

To verify proper operation of VPLS, use the following commands:

- `clear vpls mac-address instance instance-name`
- `show interfaces terse`
- `show route forwarding-table family mpls`
- `show route forwarding-table family vpls (destination | extensive | matching | table)`
- `show route instance (detail)`
- `show system statistics vpls`
- `show vpls connections`
- `show vpls statistics`

The following section shows the output of these commands on Router PE1 as a result of the configuration example:

```

user@PE1> show interfaces terse
Interface          Admin Link Proto Local          Remote
so-1/1/0           up    up
so-1/1/0.0         up    up   inet  10.11.1.5/30
                   mp1s
so-1/1/1           up    up
so-1/1/2           up    up
so-1/1/3           up    up
fe-0/1/0           up    up
fe-0/1/0.0        up   up   vp1s   # This is the local Fast Ethernet
interface.
fe-0/1/1           up    up
fe-0/1/2           up    up
fe-0/1/3           up    up
gr-0/3/0           up    up
ip-0/3/0           up    up
mt-0/3/0           up    up
pd-0/3/0           up    up
pe-0/3/0           up    up
vt-0/3/0           up    up
vt-0/3/0.32770    up   up   # This is the dynamically generated virtual port.
dsc                up    up
fxp0               up    up
fxp0.0             up    up   inet  192.186.14.218/24
fxp1               up    up
fxp1.0             up    up   tnp   4
gre                up    up
ipip               up    up
lo0                up    up
lo0.0              up    up   inet  10.245.14.218      --> 0/0
                                   127.0.0.1         --> 0/0
                                   inet6 fe80::2a0:a5ff:fe28:13e0
                                   feee::10:245:14:218

lsi                up    up
mtun               up    up
pimd               up    up
pime               up    up
tap                up    up

user@PE1> show system statistics vp1s
vp1s:
  0 total packets received
  0 with size smaller than minimum
  0 with incorrect version number
  0 packets for this host

  0 packets with no logical interface
  0 packets with no family
  0 packets with no route table
  0 packets with no auxiliary table
  0 packets with no corefacing entry
  0 packets with no CE-facing entry

  6 mac route learning requests   # This indicates that VPLS is working.
  6 mac routes learnt
  0 mac routes aged
  0 mac routes moved

```

To display VPLS source and destination MAC address accounting information, use the `destination`, `extensive`, `matching`, or `table` option with the `show route forwarding-table family vpls` command. When you analyze the display output, keep in mind the following:

- VPLS MAC address accounting is handled on a per-MAC address basis for each VPLS instance. All information is retrieved from MAC address entries in the MAC address table. VPLS MAC address accounting is performed only on local CE routers.
- The VPLS counters for source and destination MAC addresses increment continuously until the oldest MAC address entries are removed from the memory buffer, either when the entries time out or if the VPLS instance is restarted.

```

user@PE1> show route forwarding-table family vpls extensive
Routing table: green.vpls [Index 2]
VPLS:

Destination: default
  Route type: dynamic           Route reference: 0
  Flags: sent to PFE
  Next-hop type: flood          Index: 353      Reference: 1

Destination: default
  Route type: permanent        Route reference: 0
  Flags: none
  Next-hop type: discard        Index: 298      Reference: 1

Destination: fe-0/1/0.0
  Route type: dynamic           Route reference: 0
  Flags: sent to PFE
  Next-hop type: flood          Index: 355      Reference: 1

Destination: bb:bb:bb:bb:bb:bb/48 # This MAC address belongs to remote CE2.
  Route type: dynamic           Route reference: 0
  Flags: sent to PFE, prefix load balance
  Next-hop type: indirect        Index: 351      Reference: 4
  Next-hop type: Push 800000, Push 100002(top)
  Next-hop interface: so-1/1/0.0

Destination: aa:aa:aa:aa:aa:aa/48 # This MAC address belongs to local CE1.
  Route type: dynamic           Route reference: 0
  Flags: sent to PFE, prefix load balance
  Next-hop type: unicast         Index: 354      Reference: 2
  Next-hop interface: fe-0/1/0.0

user@PE1> show route forwarding-table family vpls
Routing table: green.vpls
VPLS:
Destination      Type RtRef Next hop          Type Index NhRef Netif
default          dynm  0         # This MAC address belongs to remote CE2.
                 0         indr  351      4
fe-0/1/0.0      dynm  0         Push 800000, Push 100002(top)
bb:bb:bb:bb:bb:bb/48
                 dynm  0
so-1/1/0.0
aa:aa:aa:aa:aa:aa/48 # This MAC address belongs to local CE1.
                 dynm  0         ucst  354      2 fe-0/1/0.0

```

```

user@PE1> show route forwarding-table family mpls
Routing table: mpls
MPLS:
Destination          Type RtRef Next hop          Type Index NhRef Netif
default              perm  0
0                    user  0
1                    user  0
2                    user  0
100000               user  0 10.11.1.6          swap 100001 so-1/1/0.0
800002               user  0                    Pop          vt-0/3/0.32770
vt-0/3/0.32770 (VPLS)
                    user  0                    indr  351    4
                    Push 800000, Push 100002(top)
so-1/1/0.0

```

```

user@PE1> show route instance green detail
green:
Router ID: 0.0.0.0
Type: vpls          State: Active
Interfaces:
  fe-0/1/0.0          # This is the local Fast Ethernet interface.
  vt-0/3/0.32770     # This is the dynamically generated VPLS virtual port.
Route-distinguisher: 10.245.14.218:1
Vrf-import: [ __vrf-import-green-internal__ ]
Vrf-export: [ __vrf-export-green-internal__ ]
Vrf-import-target: [ target:11111:1 ]
Vrf-export-target: [ target:11111:1 ]
Tables:
  green.l2vpn.0      : 2 routes (2 active, 0 holddown, 0 hidden)

```

```

user@PE1> show vpls connections
L2VPN Connections:

```

```

Legend for connection status (St)
OR -- out of range          WE -- intf encaps != instance encaps
EI -- encapsulation invalid Dn -- down
EM -- encapsulation mismatch VC-Dn -- Virtual circuit down
CM -- control-word mismatch -> -- only outbound conn is up
CN -- circuit not present   <- -- only inbound conn is up
OL -- no outgoing label     Up -- operational
NC -- intf encaps not CCC/TCC XX -- unknown
NP -- interface not present

```

```

Legend for interface status
Up -- operational
Dn -- down

```

**Instance: green**

**Local site: greenPE1 (1)**

```

connection-site      Type St      Time last up      # Up trans
2                    rmt  Up      Jan 24 06:26:49 2003      1

```

```

Local interface: vt-0/3/0.32770, Status: Up, Encapsulation: VPLS
Remote PE: 10.245.14.219, Negotiated control-word: No
Incoming label: 800002, Outgoing label: 800000

```

```

user@PE1> show system statistics vpls
vpls:
    0 total packets received
    0 with size smaller than minimum
    0 with incorrect version number
    0 packets for this host

    0 packets with no logical interface
    0 packets with no family
    0 packets with no route table
    0 packets with no auxiliary table
    0 packets with no corefacing entry
    0 packets with no CE-facing entry

    7 mac route learning requests
    7 mac routes learnt
    0 mac routes aged
    0 mac routes moved

user@PE1> show route instance green detail
green:
  Router ID: 0.0.0.0
  Type: vpls                State: Active
  Interfaces:
    fe-0/1/0.0
    vt-0/3/0.32770
  Route-distinguisher: 10.245.14.218:1
  Vrf-import: [ __vrf-import-green-internal__ ]
  Vrf-export: [ __vrf-export-green-internal__ ]
  Vrf-import-target: [ target:11111:1 ]
  Vrf-export-target: [ target:11111:1 ]
  Tables:
    green.l2vpn.0           : 2 routes (2 active, 0 holddown, 0 hidden)

user@PE1> show vpls statistics
Layer-2 VPN Statistics:

Instance: green

  Local interface: fe-0/1/0.0, Index: 351
  Remote provider edge router: 10.245.14.219
  Multicast packets:                363
  Multicast bytes :                  30956
  Flood packets :                    0
  Flood bytes :                      0

  Local interface: vt-0/3/0.32770, Index: 354
  Remote provider edge router: 10.245.14.219
  Multicast packets:                135
  Multicast bytes :                 12014
  Flood packets :                   135
  Flood bytes :                     12014

```

To clear all MAC address entries for a VPLS instance from the VPLS table, issue the `clear vpls mac-address instance instance-name` command. Add the `logical-router logical-router-name` option to clear entries in a VPLS instance within a logical router. Use the `mac-address` option to remove individual MAC addresses.

## For More Information

---

For additional information about VPLS, see the following:

- *JUNOS VPNs Configuration Guide*
- *JUNOS Network Interfaces Configuration Guide*
- *JUNOS Class of Service Configuration Guide*
- *JUNOS Routing Protocols Configuration Guide*
- Internet draft draft-ietf-l2vpn-vpls-bgp-08.txt, *Virtual Private LAN Service (VPLS) Using BGP for Auto-discovery and Signaling* (except section 3.4.2, expires December 2006)
- RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*

## Revision History

---

- 15 September 2006—8.1R1 Release. Richard Hendricks.
- 29 June 2006—Added limitations for MSTP and logical routers, 8.0R1 Release. Richard Hendricks.
- 27 March 2006—Added support for VPLS on LSI logical interfaces, 7.6R1 Release. Richard Hendricks.
- 9 January 2006—Added support for multihoming a CE router to multiple PE routers, 7.5R1 Release. Richard Hendricks.
- 14 September 2005—7.4R1 Release. Richard Hendricks.
- 13 June 2005—Added VPLS per-packet load balancing, support for limiting MAC address learning per interface in a VPLS domain, and migration to the new VPLS and Layer 2 VPN `signaling` statement at the `[edit protocols bgp groups group-name family l2vpn]` hierarchy level, 7.3R1 Release. Richard Hendricks.
- 5 April 2005—7.2R1 Release. Richard Hendricks.
- 2 February 2005—7.1R1 Release. Richard Hendricks.
- 6 October 2004—7.0R1 Release. Richard Hendricks.
- 6 July 2004—Added support for Ethernet VPLS over ATM LLC interface encapsulation on T-series and M320 routing platforms, the `show vpls statistics` command, and manual selection of tunnel-enabled PICs used to provide virtual ports for VPLS operation, 6.4R1 Release. Richard Hendricks.
- 5 April 2004—Updated `lt` interface families and encapsulation types and added new commands to clear MAC addresses from the VPLS table and modify the VPLS table timeout intervals, 6.3R1 Release. Richard Hendricks.
- 21 January 2004—Added new PIC support for VPLS. Richard Hendricks.
- 22 December 2003—Added VPLS CoS, VPLS graceful restart, VPLS interinstance bridging and routing, VPLS support on the T-series routing platforms, and operational mode commands for VPLS source and destination MAC accounting, 6.2R1 Release. Richard Hendricks.
- 22 September 2003—Added VPLS policers and filters, 6.1R1 Release. Richard Hendricks.
- 30 June 2003—Added the `ether-vpls-over-atm-llc` interface encapsulation type and LSP selection for VPLS instances, 6.0R1 Release. Elizabeth Lichtenberg and Richard Hendricks.
- 2 April 2003—Initial document written, 5.7R1. Richard Hendricks.