JUNIPER NETWORKS

Network Configuration Example

Validating a BGP-Based VPLS Multihoming Configuration

Release
11.4

Published: 2011-11-08

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Network Configuration Example Validating a BGP-Based VPLS Multihoming Configuration
Release 11.4

Revision History
October 2011—R1 Junos OS 11.4

The information in this document is current as of the date listed in the revision history.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

Introduction

This document provides an overview of the virtual private LAN service (VPLS) and a sample topology of a multihomed VPLS network. It also includes a configuration example and steps to validate a multihomed VPLS connection using the Junos operating system (Junos OS) command-line interface and SNMP.

Understanding the Virtual Private LAN Service

The virtual private LAN service (VPLS) is an Ethernet-based point-to-multipoint Layer 2 virtual private network (VPN) that enables you to connect geographically dispersed Ethernet local area network (LAN) sites to each other across an MPLS backbone. For customers who implement VPLS, all sites appear to be in the same Ethernet LAN even though traffic travels across service provider networks.

VPLS, in its implementation and configuration, has much in common with a Layer 2 VPN. In VPLS, a packet originating within a service provider customer's network is sent first to a customer edge (CE) device (for example, a router or Ethernet switch). It is then sent to a provider edge (PE) router within the service provider network. The packet traverses the service provider network over an MPLS label-switched path (LSP). It arrives at the egress PE router, which then forwards the traffic to the CE device at the destination customer site. The difference is that, for VPLS, packets can traverse the service provider networks in point-to-multipoint fashion, meaning that a packet originating from a CE device can be broadcast to all the PE routers participating in a VPLS routing instance. In contrast, a Layer 2 VPN forwards packets in point-to-point fashion. The paths carrying VPLS traffic between each PE router participating in a routing instance are called pseudowires.

The pseudowires are signaled using either BGP or LDP, based on the VPLS implementation. There are two standardized VPLS implementations supported by the IETF: RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*, and RFC 4762, *Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*.

RFC 4761-based implementations use BGP for auto discovery and signaling, whereas RFC 4762-based implementations use LDP for signaling (auto discovery is not supported in this model).

VPLS multihoming enables you to connect a customer site to multiple PE routers to provide redundant connectivity while preventing the formation of Layer 2 loops in the service provider network. A VPLS site that is multihomed to two or more PE routers provides redundant connectivity in the event of a PE router-to-CE device link failure or the failure of a PE router.

When multihoming a VPLS site (potentially in different autonomous systems [ASs]), the PE routers connected to the same site can either be configured with the same VPLS edge (VE) device identifier or with different VE device identifiers. If you are using different VE device identifiers, you must run the Spanning Tree Protocol (STP) on the CE device, and possibly on the PE routers, to construct a loop-free VPLS topology.

If the PE routers are connected to the same site and assigned the same VE device identifier, a loop-free topology is constructed using a routing mechanism such as BGP path selection. When a BGP speaker receives two equivalent network layer reachability information (NLRI) advertisements, it applies standard path selection criteria, such as local preference and AS path length, to determine which NLRI to choose, and selects only one.

Because a PE router picks one of the received NLRI advertisements with a particular VE device identifier, it establishes the pseudowire only to the PE router from which the winning advertisement originated. This prevents the creation of multiple paths between sites and formation of Layer 2 loops in the network. If the selected PE router fails, all PE routers in the network automatically switch to the backup PE router and establish pseudowires through the backup PE router.

Two VPLS NLRIs are considered equivalent from a path selection perspective if the following are the same:

- Route distinguisher
- VE device identifier
- VE block offset

If two PE routers are assigned the same VE device identifier in a given VPLS, they must also advertise the same VE block size for a given VE offset. The PE routers can be configured with the same route distinguisher or with distinct route distinguishers.

We recommend that you configure distinct route distinguishers for each multihomed router. Configuring distinct route distinguishers provides faster convergence when the connection to a primary router goes down. Configuring distinct route distinguishers also requires the other PE routers to maintain additional state information.

Example: Configuring a Multihomed VPLS

This example shows how to set up a multihomed VPLS network.

Requirements

The network topology discussed in this example uses the following devices:

- EX Series Ethernet Switches that are running Junos OS Release 9.4 or later for customer edge devices
- M Series Multiservice Edge Routers that are running Junos OS Release 9.3 or later for provider edge routers

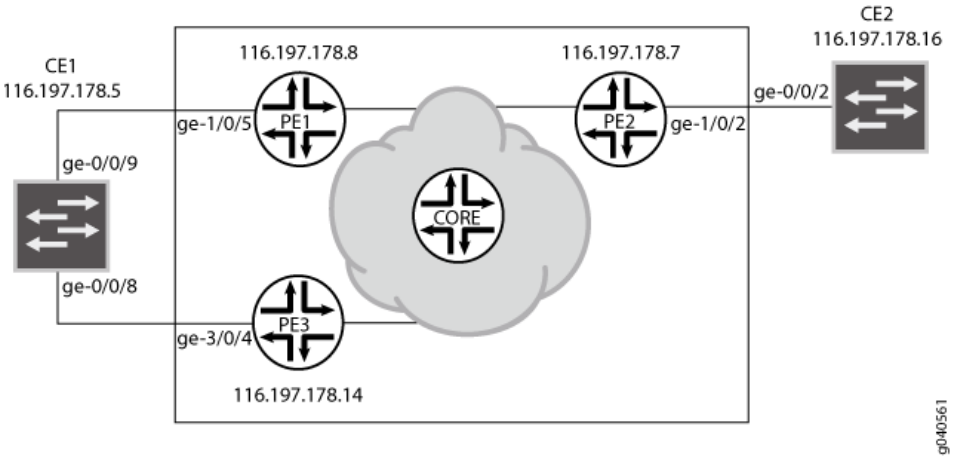
Overview

You can configure multihoming for any routing instance of type VPLS. This example provides a sample topology of a multihomed VPLS, and explains how to set up a multihomed VPLS network. This example also shows how to verify a multihomed VPLS network configuration.

Sample Topology

In this example, the customer edge device, CE1, connects over interfaces **ge-0/0/9** and **ge-0/0/8** to interfaces **ge-1/0/5** and **ge-3/0/4** on provider edge routers PE1 and PE3, respectively, to create a multihomed VPLS with Device CE2 that is connected through provider edge router PE2.

Figure 1: Sample Topology of a Multihomed VPLS Site



Guidelines for Configuring a Multihomed VPLS Site

When setting up a multihomed VPLS configuration, you must:

- Include the **multihoming** statement to enable multihoming for the site. In this example, multihoming is configured on Routers PE1 and PE3, the provider edge routers that connect to the customer edge device CE1.
- Assign the same site ID on all PE routers that are connected to the same CE device. In this example, Routers PE1 and PE3, the provider edge routers that connect to Device CE1, have the same site name (**site 1**) and site ID (**10**).
- Configure the same VLAN ID on all devices that are part of the multihomed VPLS network. In this example, VLAN ID is constant (**600**) across the customer edge devices, CE1 and CE2, and the provider edge routers, PE1, PE2, and PE3.
- Assign distinct route distinguishers for multihomed PE routers. In this example, Routers PE1, PE2, and PE3 have unique route identifiers configured on them.
- Configure the multihomed provider edge routers as primary and backup. In this example, Router PE1 is configured as the primary site, whereas Router PE3 is configured as the backup site.
- Reference all interfaces assigned to the multihomed VPLS site on each PE router. Only one of these interfaces is used to send and receive traffic for this site at a time.
- Either designate a primary interface or allow the router to select the interface to be used as the primary interface. If the router selects the interface, the interface used to connect the PE router to the site depends on the order in which interfaces are listed in the PE router's configuration. The first operational interface in the set of configured interfaces is chosen to be the designated interface. If this interface fails, the next interface in the list is selected to send and receive traffic for the site.

To enable VPLS multihoming, you must include the following configuration statements in the provider edge router configuration:

```
[edit routing-instances routing-instance-name]
instance-type vpls;
interface interface-name;
protocols {
   vpls {
   site site-name {
      active-interface {
         any;
         primary interface-name;
      }
      interface interface-name;
      multihoming;
      site-identifier number;
   }
}
route-distinguisher (as-number:id | ip-address:id);
```

Configuration

To configure a multihomed VPLS network, perform these tasks:

Configuring Customer Edge Device CE1

Step-by-Step Procedure

To configure Device CE1:

1. Enable the interfaces on which you want to configure the VPLS routing instance.

   ```
   user@CE1# set interfaces ge-0/0/8 enable
   user@CE1# set interfaces ge-0/0/9 enable
   ```

2. Enable the interfaces to receive and forward frames with 802.1Q VLAN tags.

   ```
   user@CE1# set interfaces ge-0/0/8 vlan-tagging
   user@CE1# set interfaces ge-0/0/9 vlan-tagging
   ```

3. Configure the logical interfaces, VLAN identifier value for 802.1Q VLAN tags, and the address family.

   ```
   user@CE1# set interfaces ge-0/0/8 unit 600 vlan-id 600 family inet address
     15.11.3.2/24
   user@CE1# set interfaces ge-0/0/9 unit 600 vlan-id 600 family inet address
     15.11.3.1/24
   ```

Results

After you complete the steps in the preceding procedure, the relevant section of the configuration on Device CE1 reads as:

```
interfaces ge-0/0/8 {
    enable;
    vlan-tagging;
    unit 600 {
        vlan-id 600;
        family inet {
            address 15.11.3.2/24;
        }
    }
}
interfaces ge-0/0/9 {
    enable;
    vlan-tagging;
    unit 600 {
        vlan-id 600;
        family inet {
            address 15.11.3.1/24;
        }
    }
```

```
}
```

Configuring Customer Edge Device CE2

Step-by-Step Procedure

To configure Device CE2:

1. Enable the interface on which you want to configure the VPLS routing instance.

   ```
   user@CE1# set interfaces ge-0/0/2 enable
   ```

2. Enable the interface to receive and forward frames with 802.1Q VLAN tags.

   ```
   user@CE1# set interfaces ge-0/0/2 vlan-tagging
   ```

3. Configure the logical interface, VLAN identifier value for 802.1Q VLAN tags, and the address family.

   ```
   user@CE1# set interfaces ge-0/0/2 unit 600 vlan-id 600 family inet address
     15.11.3.3/24
   ```

Results

After you complete the steps in the preceding procedure, the relevant section of configuration on Device CE2 reads as:

```
interfaces ge-0/0/2 {
   enable;
   vlan-tagging;
   unit 600 {
      vlan-id 600;
      family inet {
         address 15.11.3.3/24;
      }
   }
}
```

Configuring Provider Edge Router PE1

Step-by-Step Procedure

To configure Router PE1 for multihoming:

1. Configure the interface properties.

   ```
   [edit interfaces ge-1/0/5]
   user@PE1# set enable
   user@PE1# set flexible-vlan-tagging
   user@PE1# set encapsulation vlan-vpls
   user@PE1# set unit 10

   [edit interfaces ge-1/0/5 unit 10]
   user@PE1# set encapsulation vlan-vpls
   user@PE1# set vlan-id 600
   user@PE1# set family vpls
   ```

2. Create a routing instance of type VPLS, and configure the following parameters.

   ```
   [edit]
   user@PE1# set routing-instances vpls-multihoming instance-type vpls

   [edit routing-instances vpls-multihoming]
   ```

```
user@PE1# set interface ge-1/0/5.10
user@PE1# set route-distinguisher 65410:102
user@PE1# set vrf-target target:65410:0

[edit routing-instances vpls-multihoming protocols vpls]
user@PE1# set no-tunnel services

[edit routing-instances vpls-multihoming protocols vpls site site1]
user@PE1# set site identifier 10
user@PE1# set multihoming
user@PE1# set site-preference primary
user@PE1# set interface ge-1/0/5.10
user@PE1# set active-interface primary ge-1/0/5.10
```

3. Configure the routing options.

```
[edit routing-options]
user@PE1# set static route 0.0.0.0/0 next-hop [116.197.178.110.93.54.254]
user@PE1# set router-id 192.1.1.4
user@PE1# set autonomous-system 69
user@PE1# set forwarding-table export exp-to-fwd
```

4. Configure BGP properties.

```
[edit protocols bgp]
user@PE1# set group vpls-pe type internal

[edit protocols bgp group vpls-pe]
user@PE1# set local- address 192.1.1.4
user@PE1# set family l2vpn signaling
user@PE1# set neighbor 192.1.1.5
user@PE1# set neighbor 192.1.1.6
user@PE1# set neighbor 192.1.1.3
```

5. Configure RSVP properties.

```
[edit]
user@PE1# set protocols rsvp interface all aggregate
```

6. Configure OSPF properties.

```
[edit protocols ospf]
user@PE1# set traceoptions file ospf world-readable flag error
user@PE1# set overload timeout 600
user@PE1# set traffic-engineering
user@PE1# set lsa-refresh-interval 30

[edit protocols ospf area 0.0.0.0]
user@PE1# set interface lo.0.0 passive
user@PE1# set interface ge-2/2/2.0 metric 11
```

7. Configure policy options.

```
[edit]
user@PE1# set policy-options policy-statement exp-to-fwd

[edit policy-options policy-statement exp-to-fwd]
user@PE1# set term a
```

```
[edit policy-options policy-statement exp-to-fwd term a]
user@PE1# set from community test
user@PE1# set then install next-nexthop lsp [pe1-to-pe3 pe1-to-pe2]
user@PE1# set then accept
user@PE1# exit
user@PE1# set policy-options policy-statement next-hop-self

[edit policy-options policy-statement next-hop-self]
user@PE1# set then next-hop-self
user@PE1# set then next policy
user@PE1# exit
user@PE1# set policy-options community test members target:65410:0
```

Results After you complete this procedure, the configuration on Router PE1 reads:

```
routing-instances vpls-multihoming instance-type vpls {
   interface ge-1/0/5.10;
   route-distinguisher 65410:102;
   vrf-target target:65410:0;
   protocols {
      vpls {
         no-tunnel-services;
         site site1 {
            site-identifier 10;
            multihoming;
            site-preference primary;
            active-interface primary ge-1/0/5.10;
            interface ge-1/0/5.10;
         }
      }
   }
}
routing-options {
   static {
      route 0.0.0.0/0 next-hop [ 116.197.178.1 10.93.54.254 ];
   }
   router-id 192.1.1.4;
   autonomous-system 69;
   forwarding-table {
      export exp-to-fwd;
   }
}
interfaces ge-1/0/5 {
   enable;
   flexible-vlan-tagging;
   encapsulation vlan-vpls;
   unit 10 {
      encapsulation vlan-vpls;
      vlan-id 600;
      family vpls;
   }
}
protocols{
   bgp {
      group vpls-pe {
```



```
        type internal;
        local-address 192.1.1.4;
        family l2vpn {
          signaling;
        }
        neighbor 192.1.1.5;
        neighbor 192.1.1.6;
        neighbor 192.1.1.3;
      }
    }
    rsvp {
      interface all {
        aggregate;
      }
    }
    ospf {
      traceoptions {
        file ospf world-readable;
        flag error;
      }
      overload timeout 600;
      traffic-engineering;
      lsa-refresh-interval 30;
      area 0.0.0.0 {
        interface lo0.0 {
          passive;
        }
        interface ge-2/2/2.0 {
          metric 11;
        }
      }
    }
}
policy-options {
    policy-statement exp-to-fwd {
      term a {
        from community test;
        then {
          install-nexthop lsp [ pe1-to-pe3 pe1-to-pe2 ];
          accept;
        }
      }
    }
    policy-statement next-hop-self {
      then {
        next-hop self;
        next policy;
      }
    }
    community test members target:65410:0;
}
```

Configuring Provider Edge Router PE2

Step-by-Step Procedure

To configure Router PE2 for multihoming:

1. Configure the interface properties.

```
[edit interfaces ge-1/0/2]
user@PE2# set enable
user@PE2# set flexible-vlan-tagging
user@PE2# set encapsulation flexible-ethernet-services
user@PE2# set unit 10

[edit interfaces ge-1/0/5 unit 10]
user@PE2# set encapsulation vlan-vpls
user@PE2# set vlan-id 600
user@PE2# set family vpls
```

2. Create a routing instance of type VPLS, and configure the following parameters.

```
[edit]
user@PE2# set routing-instances vpls-multihoming instance-type vpls

[edit routing-instances vpls-multihoming]
user@PE2# set interface ge-1/0/1.10
user@PE2# set interface ge-1/0/2.100
user@PE2# set route-distinguisher 65410:103
user@PE2# set vrf-target target:65410:0

[edit routing-instances vpls-multihoming protocols vpls]
user@PE2# set no-tunnel services

[edit routing-instances vpls-multihoming protocols vpls site site2]
user@PE2# set site identifier 20
```

3. Configure the routing options.

```
[edit routing-options]
user@PE2# set static route 0.0.0.0/0 next-hop [116.197.178.1 10.93.54.254]
user@PE2# set static route 6.6.6.8/32 next-hop 116.197.179.15
user@PE2# set static route 18.1.1.1/32 next-hop 116.197.179.15
user@PE2# set router-id 192.1.1.3
user@PE2# set autonomous-system 69
user@PE2# set forwarding-table export exp-to-fwd
```

4. Configure BGP properties.

```
[edit protocols bgp]
user@PE2# set group vpls-pe type internal

[edit protocols bgp group vpls-pe]
user@PE2# set local- address 192.1.1.3
user@PE2# set family l2vpn signaling
user@PE2# set neighbor 192.1.1.4
user@PE2# set neighbor 192.1.1.5
user@PE2# set neighbor 192.1.1.6
```

5. Configure RSVP properties.

```
[edit]
user@PE2# set protocols rsvp interface all aggregate
```

6. Configure OSPF properties.

```
[edit protocols ospf]
user@PE2# set traceoptions file ospf world-readable flag error
user@PE2# set overload timeout 600
user@PE2# set traffic-engineering
user@PE2# set lsa-refresh-interval 30

[edit protocols ospf area 0.0.0.0]
user@PE2# set interface lo.0.0 passive
user@PE2# set interface ge-2/2/1.0 metric 12
```

7. Configure policy options.

```
[edit]
user@PE2# set policy-options policy-statement exp-to-fwd

[edit policy-options policy-statement exp-to-fwd]
user@PE2# set term a

[edit policy-options policy-statement exp-to-fwd term a]
user@PE2# set from community test
user@PE2# set then install next-nexthop lsp [pe2-to-PE1 pe2-to-pe3]
user@PE2# set then accept
user@PE2# exit
user@PE2# set policy-options policy-statement next-hop-self

[edit policy-options policy-statement next-hop-self]
user@PE2# set then next-hop-self
user@PE2# set then next policy
user@PE2# exit
user@PE2# set policy-options community test members target:65410:0
```

Results After you complete this procedure, the configuration on Router PE2 reads:

```
interfaces ge-1/0/2 {
   enable;
   flexible-vlan-tagging;
   encapsulation flexible-ethernet-services;
   unit 10 {
      encapsulation vlan-vpls;
      vlan-id 600;
      family vpls;
   }
}
routing-instances vpls-multihoming {
   instance-type vpls;
   interface ge-1/0/1.10;
   interface ge-1/0/2.100;
   route-distinguisher 65410:103;
   vrf-target target:65410:0;
```

```
    protocols {
        vpls {
            no-tunnel-services;
            site site2 {
                site-identifier 20;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 next-hop [ 116.197.178.1 10.93.54.254 ];
        route 6.6.6.8/32 next-hop 116.197.179.15;
        route 18.1.1.1/32 next-hop 116.197.179.15;
    }
    router-id 192.1.1.3;
    autonomous-system 69;
    forwarding-table {
        export exp-to-fwd;
    }
}
protocols {
    bgp {
        group vpls-pe {
            type internal;
            local-address 192.1.1.3;
            family l2vpn {
                signaling;
            }
            neighbor 192.1.1.4;
            neighbor 192.1.1.5;
            neighbor 192.1.1.6;
        }
    }
    rsvp {
        interface all {
            aggregate;
        }
    }
    ospf {
        traceoptions {
            file ospf world-readable;
            flag error;
        }
        overload timeout 600;
        traffic-engineering;
        lsa-refresh-interval 30;
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface ge-2/2/1.0 {
                metric 12;
            }
        }
    }
```

```
}
policy-options {
  policy-statement exp-to-fwd {
    term a {
      from community test;
      then {
        install-nexthop lsp [ pe2-to-pe1 pe2-to-pe3 ];
        accept;
      }
    }
  }
  policy-statement next-hop-self {
    then {
      next-hop self;
      next policy;
    }
  }
  community test members target:65410:0;
}
```

Configuring Provider Edge Router PE3

Step-by-Step Procedure

To configure Router PE3 for multihoming:

1. Configure the interface properties.

```
[edit interfaces ge-3/0/4]
user@PE3# set enable
user@PE3# set flexible-vlan-tagging
user@PE3# set encapsulation vlan-vpls
user@PE3# set unit 100

[edit interfaces ge-3/0/4 unit 100]
user@PE3# set encapsulation vlan-vpls
user@PE3# set vlan-id 600
user@PE3# set family vpls
```

2. Create a routing instance of type VPLS, and configure the following parameters.

```
[edit]
user@PE3# set routing-instances vpls-multihoming instance-type vpls

[edit routing-instances vpls-multihoming]
user@PE3# set interface ge-3/0/4.100
user@PE3# set route-distinguisher 65410:100
user@PE3# set vrf-target target:65410:0

[edit routing-instances vpls-multihoming protocols vpls]
user@PE3# set no-tunnel services

[edit routing-instances vpls-multihoming protocols vpls site site1]
user@PE3# set site identifier 10
user@PE3# set multihoming
user@PE3# set site-preference backup
```

3. Configure the routing options.

```
[edit routing-options]
user@PE3# set static route 0.0.0.0/0 next-hop 116.197.178.1
user@PE3# set router-id 192.1.1.6
user@PE3# set autonomous-system 69
user@PE3# set forwarding-table export exp-to-fwd
```

4. Configure BGP properties.

```
[edit protocols bgp]
user@PE3# set group vpls-pe type internal

[edit protocols bgp group vpls-pe]
user@PE3# set local- address 192.1.1.6
user@PE3# set family l2vpn signaling
user@PE3# set neighbor 192.1.1.4
user@PE3# set neighbor 192.1.1.5
user@PE3# set neighbor 192.1.1.3
```

5. Configure RSVP properties.

```
[edit]
user@PE3# set protocols rsvp interface all aggregate
```

6. Configure OSPF properties.

```
[edit protocols ospf]
user@PE3# set traceoptions file ospf world-readable flag error
user@PE3# set overload timeout 600
user@PE3# set traffic-engineering
user@PE3# set lsa-refresh-interval 30

[edit protocols ospf area 0.0.0.0]
user@PE3# set interface lo.0.0 passive
user@PE3# set interface ge-3/0/0.0 metric 15
```

7. Configure policy options.

```
[edit]
user@PE3# set policy-options policy-statement exp-to-fwd

[edit policy-options policy-statement exp-to-fwd]
user@PE3# set term a

[edit policy-options policy-statement exp-to-fwd term a]
user@PE3# set from community test
user@PE3# set then install next-nexthop lsp [pe3-to-pe1 pe3-to-pe2]
user@PE3# set then accept
user@PE3# exit
user@PE3# set policy-options policy-statement next-hop-self

[edit policy-options policy-statement next-hop-self]
user@PE3# set then next-hop-self
user@PE3# set then next policy
user@PE3# exit
user@PE3# set policy-options community test members target:65410:0
```

Results After you complete this procedure, the configuration on Router PE3 reads:

```
interfaces ge-3/0/4 {
    enable;
    encapsulation vlan-vpls;
    flexible-vlan-tagging;
    unit 100 {
        encapsulation vlan-vpls;
        vlan-id 600;
        family vpls;
    }
}
routing-instances vpls-multihoming instance-type vpls {
    interface ge-3/0/4.100;
    route-distinguisher 65410:100
    vrf-target target:65410:0;
    protocols {
        vpls {
            no-tunnel-services;
            site site1 {
                site-identifier 10;
                multihoming;
                site-preference backup;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 next-hop 116.197.178.1;
    }
    router-id 192.1.1.6;
    autonomous-system 69;
    forwarding-table {
        export exp-to-fwd;
    }
}
protocols {
    bgp {
        group vpls-pe {
            type internal;
            local-address 192.1.1.6;
            family l2vpn {
                signaling;
            }
            neighbor 192.1.1.4;
            neighbor 192.1.1.5;
            neighbor 192.1.1.3;
        }
    }
    ospf {
        traceoptions {
            file ospf world-readable;
            flag error;
        }
        overload timeout 600;
```

```
        traffic-engineering;
        lsa-refresh-interval 30;
        area 0.0.0.0 {
          interface ge-3/0/0.0 {
            metric 15;
          }
          interface lo0.0 {
            passive;
          }
        }
      }
      rsvp {
        interface all {
          aggregate;
        }
      }
    }
    policy-options {
      policy-statement exp-to-fwd {
        term a {
          from community test;
          then {
            install-nexthop lsp [ pe3-to-pe1 pe3-to-pe2 ];
            accept;
          }
        }
      }
      policy-statement next-hop-self {
        then {
          next-hop self;
          next policy;
        }
      }
      community test members target:65410:0;
    }
```

Verification

Confirm that the configuration is working properly.

- Verifying a Multihomed VPLS Connection from the Junos OS CLI on page 18

Verifying a Multihomed VPLS Connection from the Junos OS CLI

Purpose Validate the multihomed VPLS connection.

Action Verify the multihomed VPLS connections on the provider edge routers.

1. Enter the **show vpls connections** command on Router PE1.

```
user@PE1> show vpls connections

Layer-2 VPN connections:

Legend for connection status (St)
EI -- encapsulation invalid      NC -- interface encapsulation not CCC/TCC/VPLS
```

```
EM -- encapsulation mismatch     WE -- interface and instance encaps not same
VC-Dn -- Virtual circuit down    NP -- interface hardware not present
CM -- control-word mismatch      -> -- only outbound connection is up
CN -- circuit not provisioned    <- -- only inbound connection is up
OR -- out of range               Up -- operational
OL -- no outgoing label          Dn -- down
LD -- local site signaled down   CF -- call admission control failure
RD -- remote site signaled down  SC -- local and remote site ID collision
LN -- local site not designated  LM -- local site ID not minimum designated
RN -- remote site not designated RM -- remote site ID not minimum designated
XX -- unknown connection status  IL -- no incoming label
MM -- MTU mismatch               MI -- Mesh-Group ID not availble
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy

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

Instance: vpls-multihoming
  Local site: site1 (10)
    connection-site      Type   St     Time last up      # Up trans
    10                   rmt   RN
    20                   rmt   Up     May 14 07:54:42 2010    1
      Remote PE: 192.1.1.3, Negotiated control-word: No
      Incoming label: 262164, Outgoing label: 262154
      Local interface: lsi.1048834, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls vpls-multihoming local site 10 remote site 20
```

The output shows that Router PE1 is connected to two sites (ID: 10 and 20), where site 10 is a remote site that is not designated, and site 20 is the remote site that is connected to Router PE1.

2. Enter the **show vpls connections** command on Router PE2.

```
user@PE2> show vpls connections

Layer-2 VPN connections:

~
~
~
Instance: vpls-multihoming
  Local site: site2 (20)
  connection-site         Type  St     Time last up       # Up trans
  10                      rmt   Up     May 14 07:29:55 2010    1
      Remote PE: 192.1.1.4, Negotiated control-word: No
      Incoming label: 262154, Outgoing label: 262164
      Local interface: lsi.1048576, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls vpls-multihoming local site 20 remote site 10
```

The output shows that Router PE2, with site ID 20, is connected to the remote site (192.1.1.4) with site ID 10, that is, Router PE1.

3. Enter the **show vpls connections** command on Router PE3.

```
user@PE3> show vpls connections

Layer-2 VPN connections:

~
```

```
~
~
Instance: vpls-multihoming
  Local site: site1 (10)
    connection-site           Type  St     Time last up          # Up trans
    10                        rmt   LN
    20                        rmt   LN
```

The output shows that Router PE3 is connected to two remote sites, site ID 10 and site ID 20.

Related Documentation

- Understanding the Virtual Private LAN Service on page 3
- Verifying a Multihomed VPLS Connection Using SNMP MIB Walks on page 27

SNMP MIB Support for BGP-Based VPLS

The enterprise-specific VPN MIB, whose object identifier is **jnxMibs 26**, provides SNMP MIB objects and traps for monitoring VPLS among other types of VPNs.

You can access the enterprise-specific VPN MIB from http://www.juniper.net/techpubs/en_US/junos11.2/topics/reference/ mibs/mib-jnx-vpn.txt .

For a detailed interpretation of the enterprise-specific MIB objects, see the *Junos OS SNMP MIBs and Traps Reference*.

This topic contains the following sections:

jnxVpnTable

The Juniper Networks enterprise-specific **jnxVpnTable**, whose object identifier is **{jnxVpnMibobjects 2}**, lists configured VPNs.

Each **jnxVpnEntry** contains information about a configured VPN with the objects listed in Table 1 on page 21 and their supported VPNs and circuit connection services. The first two objects in **jnxVpnEntry** (**jnxVpnType** and **jnxVpnName**) are indexes and are not included in this table.

Table 1: jnxVpnTable Objects

Object	Description
jnxVpnRowStatus	Creates, modifies, or deletes a row in this table.
jnxVpnStorageType	The storage type.
jnxVpnDescription	VPN description.
jnxVpnIdentifierType	Type of **jnxVpnIdentifer**.
jnxVpnIdentifier	For BGP VPNs, this is the route distinguisher for the VPN. For LDP VPNs, this is the virtual circuit (VC) ID for the circuit. A value of all zeros indicates that a route distinguisher and a VC ID are not configured for the VPN.
jnxVpnConfiguredSites	The number of sites configured in the VPN.
jnxVpnActiveSites	The number of active sites in the VPN.

Table 1: jnxVpnTable Objects *(continued)*

Object	Description
jnxVpnLocalAddresses	The number of addresses learned from the CE device.
jnxVpnTotalAddresses	The total number of addresses in the VPN routing table.
jnxVpnVpnAge	The age of the VPN, in hundredths of a second.

jnxVpnIfTable

The Juniper Networks enterprise-specific **jnxVpnIfTable**, whose object identifier is **{jnxVpnMibObjects 3}**, lists VPN interfaces.

Each **jnxVpnIfEntry** contains information about VPN interfaces. The objects are listed in Table 2 on page 22. The first three objects (**jnxVpnIfVpnType**, **jnxVpnIfVpnName**, and **jnxVpnIfIndex**) are indexes and are not included in this table.

Table 2: jnxVpnIfTable Objects

Object	Description
jnxVpnIfRowStatus	Creates, modifies, or deletes a row in this table.
jnxVpnIfStorageType	Identifies the storage type for an object.
jnxVpnIfAssociationPw	The index of the associated pseudowire. If no index is associated with a pseudowire, the index is 0. A pseudowire is a mechanism that carries essential elements of an emulated circuit from one provider edge (PE) device to one or more other PEs over a packet-switched network (PSN).

Table 2: jnxVpnIfTable Objects *(continued)*

Object	Description
jnxVpnIfProtocol	Indicates the protocol running over a VPN interface. This object contains the following values: • **other(0)** • **frameRelay(1)** • **atmAal5(2)** • **atmCell(3)** • **ethernetVlan(4)** • **ethernet(5)** • **ciscoHdlc(6)** • **ppp(7)** • **cem(8)** • **atmVcc(9)** • **atmVpc(10)** • **vpls(11)** • **ipInter- working(12)** • **snapInter- working(13)** • **static(20)** • **rip(21)** • **ospf(22)** • **bgp(23)** • **atmTrunkNNI (129)** • **atmTrunkUNI (130)**
jnxVpnIfInBandwidth	The maximum bandwidth that the customer edge (CE) device connected over a VPN can send to the PE device, in Kbps. A value of 0 indicates that there is no configured maximum.
jnxVpnIfOutBandwidth	The maximum bandwidth that the PE device can send to the CE device over a VPN interface, in Kbps. A value of 0 indicates that there is no configured maximum.
jnxVpnIfStatus	Status of a monitored VPN interface. This object contains the following values: • **unknown(0)** • **noLocal- Interface(1)** • **disabled(2)** • **encapsulation- Mismatch(3)** • **down(4)** • **up(5)**

jnxVpnPwTable

The Juniper Networks enterprise-specific **jnxVpnPwTable**, whose object identifier is **{jnxVpnMibObjects 4}**, lists pseudowire connections.

Each **jnxVpnPwEntry** contains pseudowire information about a VPN that is being monitored, and has the objects listed in Table 3 on page 24. The first three objects (**jnxVpnPwVpnType**, **jnxVpnPwVpnName**, and **jnxVpnPwIndex)** are indexes and are not listed in this table.

Table 3: jnxVpnPwTable Objects

Object	Description
jnxVpnPwRowStatus	Creates, modifies, and deletes a row in this table.
jnxVpnPwStorageType	The storage type.
jnxVpnPwAssociatedInterface	The VPN index of the interface associated with a pseudowire. If no interface is associated with a pseudowire, 0 is returned.
jnxVpnPwLocalSiteId	The local site identifier for a pseudowire. When there is no local site identifier, 0 is returned.
jnxVpnPwRemoteSiteId	The remote site identifier. For example, the site at the end of the pseudowire. When there is no remote site identifier, 0 is returned.
jnxVpnRemotePeIdAddrType	The remote PE address. For example, the router at the end of the pseudowire.
jnxVpnRemotePeIdAddress	The type of tunnel over which the pseudowire is carried. If several pseudowires can be carried in one tunnel, each pseudowire is identified by the multiplexer or demultiplexer within a tunnel. This object can contain the following values: • **static(1)** • **gre(2)** • **l2tpv3(3)** • **ipSec(4)** • **ldp(5)** • **rsvpTe(6)** • **crLdp(7)**
jnxVpnPwTunnelType	The type of tunnel over which the pseudowire is carried.
jnxVpnPwTunnelName	The name of the tunnel over which the pseudowire is carried.
jnxVpnPwReceiveDemux	The demultiplexer value that identifies received packets associated with this pseudowire.
jnxVpnPwTransmitDemux	The demultiplexer value that identifies the transmitted packets associated with this pseudowire.

Table 3: jnxVpnPwTable Objects *(continued)*

Object	Description
jnxVpnPwStatus	The status of the pseudowire. This object can have the following values: • **unknown(0)** • **down(1)** • **up(2)**
jnxVpnPwTunnelStatus	The status of the PE-to-PE tunnel over which the pseudowire is carried.
jnxVpnPwRemoteSiteStatus	The interface status at the remote end of the pseudowire. This object can have the following values: • **unknown(0)** • **outOfRange(1)** • **down(2)** • **up(3)**
jnxVpnPwTimeUp	The time, in hundredths of a second, that a pseudowire has been operational.
jnxVpnPwTransitions	The number of state transitions (up to down and down to up) that a tunnel has undergone.
jnxVpnPwLastTransition	The time, in hundredths of a second, since the last transition occurred in a tunnel.
jnxVpnPwPacketsSent	The number of packets sent over a pseudowire.
jnxVpnPwOctetsSent	The number of octets sent over a pseudowire.
jnxVpnPwPacketsReceived	The number of packets received over a pseudowire.
jnxVpnPwOctetsReceived	The number of octets received over a pseudowire.
jnxVpnPwLRPacketsSent	The number of packets sent over a pseudowire.
jnxVpnPwLROctetsSent	The number of octets sent over a pseudowire.
jnxVpnPwLRPacketsReceived	The number of packets received over a pseudowire.
jnxVpnPwLROctetsReceived	The number of octets received over a pseudowire.

jnxVpnRTTable

The Juniper Networks enterprise-specific **jnxVpnRTTable**, whose object identifier is **{jnxVpnMibObjects 5}** contains route targets for a VPN.

Each **jnxVpnRTEntry** lists route targets for a given VPN, and has the objects listed in Table 4 on page 26. The first three objects (**jnxVpnRTVpnType**, **jnxVpnRTVpnName**, and **jnxVpnRTIndex**) are indexes and are not listed in this table.

Table 4: jnxVpnRTTable Objects

Object	Description
jnxVpnRTRowStatus	Creates, modifies, or deletes a row in this table.
jnxVpnRTStorageType	Identifies the storage type for an object.
jnxVpnRTType	The type of the following route target. The type can be **routeTarget[012]** or **none**.
jnxVpnRT	The VPN route target. If **jnxVpnRTType** is **none**, the value must be all zeros.
jnxVpnRTFunction	The route target export distribution type.

Enterprise-Specific VPN Traps

The enterprise-specific VPN MIB provides traps for monitoring VPNs. Table 5 on page 26 lists supported VPN traps, VPNs, and circuit connection services.

Table 5: VPN Traps

Object	Description
jnxVpnIfUp	Indicates that the interface with the index **jnxVpnIfIndex** belonging to the **jnxVpnIfVpnName** of type **jnxVpnIfVpnType** came up.
jnxVpnIfDown	Indicates that the interface with index **jnxVpnIfIndex** belonging to **jnxVpnIfVpnName** of type **jnxVpnIfVpnType** went down.
jnxVpnPwUp	Indicates that the pseudowire with the index **jnxVpnPwIndex** belonging to **jnxVpnPwVpnName** of type **jnxVpnPwVpnType** came up.
jnxVpnPwDown	Indicates that the pseudowire with index **jnxVpnPwIndex** belonging to **jnxVpnPwVpnName** of type **jnxVpnPwVpnType** went down.

Related Documentation

- Verifying a Multihomed VPLS Connection Using SNMP MIB Walks on page 27

Verifying a Multihomed VPLS Connection Using SNMP MIB Walks

Purpose Monitor a BGP-based VPLS multihoming configuration.

The Junos OS also enables you to use SNMP MIB walks to validate multihomed VPLS connections. This section explains the various MIB objects and traps that help you monitor BGP-based VPLS.

The SNMP walks show values configured in "Example: Configuring a Multihomed VPLS" on page 5.

Action

```
user@PE1> show snmp mib walk jnxVpnTable
jnxVpnRowStatus.4."vpls1" = 1
jnxVpnRowStatus.4."vpls-multihoming" = 1
jnxVpnStorageType.4."vpls1" = 5
jnxVpnStorageType.4."vpls-multihoming" = 5
jnxVpnDescription.4."vpls1" = vpls1
jnxVpnDescription.4."vpls-multihoming" = vpls-multihoming
jnxVpnIdentifierType.4."vpls1" = 5
jnxVpnIdentifierType.4."vpls-multihoming" = 5
jnxVpnIdentifier.4."vpls1" =  5-65052:1
jnxVpnIdentifier.4."vpls-multihoming" =  5-65410:102
jnxVpnConfiguredSites.4."vpls1" = 0
jnxVpnConfiguredSites.4."vpls-multihoming" = 0
jnxVpnActiveSites.4."vpls1" = 0
jnxVpnActiveSites.4."vpls-multihoming" = 0
jnxVpnLocalAddresses.4."vpls1" = 0
jnxVpnLocalAddresses.4."vpls-multihoming" = 0
jnxVpnTotalAddresses.4."vpls1" = 0
jnxVpnTotalAddresses.4."vpls-multihoming" = 0
jnxVpnAge.4."vpls1" = 19395600
jnxVpnAge.4."vpls-multihoming" = 19395600

user@PE1> show snmp mib walk jnxVpnIfTable
jnxVpnIfRowStatus.4."vpls1".215 = 1
jnxVpnIfRowStatus.4."vpls1".225 = 1
jnxVpnIfRowStatus.4."vpls-multihoming".247 = 1
jnxVpnIfStorageType.4."vpls1".215 = 5
jnxVpnIfStorageType.4."vpls1".225 = 5
jnxVpnIfStorageType.4."vpls-multihoming".247 = 5
jnxVpnIfAssociatedPw.4."vpls1".215 = 0
jnxVpnIfAssociatedPw.4."vpls1".225 = 0
jnxVpnIfAssociatedPw.4."vpls-multihoming".247 = 0
jnxVpnIfProtocol.4."vpls1".215 = 11
jnxVpnIfProtocol.4."vpls1".225 = 11
jnxVpnIfProtocol.4."vpls-multihoming".247 = 11
jnxVpnIfInBandwidth.4."vpls1".215 = 0
jnxVpnIfInBandwidth.4."vpls1".225 = 0
jnxVpnIfInBandwidth.4."vpls-multihoming".247 = 0
jnxVpnIfOutBandwidth.4."vpls1".215 = 0
jnxVpnIfOutBandwidth.4."vpls1".225 = 0
jnxVpnIfOutBandwidth.4."vpls-multihoming".247 = 0
jnxVpnIfStatus.4."vpls1".215 = 5
jnxVpnIfStatus.4."vpls1".225 = 5
jnxVpnIfStatus.4."vpls-multihoming".247 = 5

user@PE1> show snmp mib walk jnxVpnPwTable
```

```
jnxVpnPwRowStatus.4."vpls1".262147 = 1
jnxVpnPwRowStatus.4."vpls-multihoming".655380 = 1
jnxVpnPwStorageType.4."vpls1".262147 = 5
jnxVpnPwStorageType.4."vpls-multihoming".655380 = 5
jnxVpnPwAssociatedInterface.4."vpls1".262147 = 269
jnxVpnPwAssociatedInterface.4."vpls-multihoming".655380 = 270
jnxVpnPwLocalSiteId.4."vpls1".262147 = 4
jnxVpnPwLocalSiteId.4."vpls-multihoming".655380 = 10
jnxVpnPwRemoteSiteId.4."vpls1".262147 = 3
jnxVpnPwRemoteSiteId.4."vpls-multihoming".655380 = 20
jnxVpnRemotePeIdAddrType.4."vpls1".262147 = 1
jnxVpnRemotePeIdAddrType.4."vpls-multihoming".655380 = 1
jnxVpnRemotePeIdAddress.4."vpls1".262147 = 192.1.1.6
jnxVpnRemotePeIdAddress.4."vpls-multihoming".655380 = 192.1.1.3
jnxVpnPwTunnelType.4."vpls1".262147 = 6
jnxVpnPwTunnelType.4."vpls-multihoming".655380 = 6
jnxVpnPwTunnelName.4."vpls1".262147 = pe1-to-pe3
jnxVpnPwTunnelName.4."vpls-multihoming".655380 = pe1-to-pe2
jnxVpnPwReceiveDemux.4."vpls1".262147 = 262171
jnxVpnPwReceiveDemux.4."vpls-multihoming".655380 = 262164
jnxVpnPwTransmitDemux.4."vpls1".262147 = 262148
jnxVpnPwTransmitDemux.4."vpls-multihoming".655380 = 262154
jnxVpnPwStatus.4."vpls1".262147 = 2
jnxVpnPwStatus.4."vpls-multihoming".655380 = 2
jnxVpnPwTunnelStatus.4."vpls1".262147 = 0
jnxVpnPwTunnelStatus.4."vpls-multihoming".655380 = 0
jnxVpnPwRemoteSiteStatus.4."vpls1".262147 = 0
jnxVpnPwRemoteSiteStatus.4."vpls-multihoming".655380 = 0
jnxVpnPwTimeUp.4."vpls1".262147 = 19357900
jnxVpnPwTimeUp.4."vpls-multihoming".655380 = 19165500
jnxVpnPwTransitions.4."vpls1".262147 = 1
jnxVpnPwTransitions.4."vpls-multihoming".655380 = 1
jnxVpnPwLastTransition.4."vpls1".262147 = 19357900
jnxVpnPwLastTransition.4."vpls-multihoming".655380 = 19165500
jnxVpnPwPacketsSent.4."vpls1".262147 = 0
jnxVpnPwPacketsSent.4."vpls-multihoming".655380 = 0
jnxVpnPwOctetsSent.4."vpls1".262147 = 0
jnxVpnPwOctetsSent.4."vpls-multihoming".655380 = 0
jnxVpnPwPacketsReceived.4."vpls1".262147 = 0
jnxVpnPwPacketsReceived.4."vpls-multihoming".655380 = 0
jnxVpnPwOctetsReceived.4."vpls1".262147 = 0
jnxVpnPwOctetsReceived.4."vpls-multihoming".655380 = 0
jnxVpnPwLRPacketsSent.4."vpls1".262147 = 0
jnxVpnPwLRPacketsSent.4."vpls-multihoming".655380 = 0
jnxVpnPwLROctetsSent.4."vpls1".262147 = 0
jnxVpnPwLROctetsSent.4."vpls-multihoming".655380 = 0
jnxVpnPwLRPacketsReceived.4."vpls1".262147 = 0
jnxVpnPwLRPacketsReceived.4."vpls-multihoming".655380 = 0
jnxVpnPwLROctetsReceived.4."vpls1".262147 = 0
jnxVpnPwLROctetsReceived.4."vpls-multihoming".655380 = 0
```

```
user@PE1> show snmp mib walk jnxVpnRTTable
jnxVpnRTRowStatus.4."vpls1".1 = 1
jnxVpnRTRowStatus.4."vpls-multihoming".1 = 1
jnxVpnRTStorageType.4."vpls1".1 = 5
jnxVpnRTStorageType.4."vpls-multihoming".1 = 5
jnxVpnRTType.4."vpls1".1 = 6
jnxVpnRTType.4."vpls-multihoming".1 = 6
jnxVpnRT.4."vpls1".1 = 65052:1
jnxVpnRT.4."vpls-multihoming".1 = 65410:0
```



```
jnxVpnRTFunction.4."vpls1".1 = 3
jnxVpnRTFunction.4."vpls-multihoming".1 = 3
```

Related Documentation

- Example: Configuring a Multihomed VPLS on page 5

