

# Network Configuration Example

## Validating a BGP-Based VPLS Multihoming Configuration

Release  
**11.2**



---

Published: 2011-05-19

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

This product includes the Envoy SNMP Engine, developed by Epilogue Technology, an Integrated Systems Company. Copyright © 1986-1997, Epilogue Technology Corporation. All rights reserved. This program and its documentation were developed at private expense, and no part of them is in the public domain.

This product includes memory allocation software developed by Mark Moraes, copyright © 1988, 1989, 1993, University of Toronto.

This product includes FreeBSD software developed by the University of California, Berkeley, and its contributors. All of the documentation and software included in the 4.4BSD and 4.4BSD-Lite Releases is copyrighted by the Regents of the University of California. Copyright © 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994. The Regents of the University of California. All rights reserved.

GateD software copyright © 1995, the Regents of the University. All rights reserved. Gate Daemon was originated and developed through release 3.0 by Cornell University and its collaborators. Gated is based on Kirton's EGP, UC Berkeley's routing daemon (routed), and DCN's HELLO routing protocol. Development of Gated has been supported in part by the National Science Foundation. Portions of the GateD software copyright © 1988, Regents of the University of California. All rights reserved. Portions of the GateD software copyright © 1991, D. L. S. Associates.

This product includes software developed by Maker Communications, Inc., copyright © 1996, 1997, Maker Communications, Inc.

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Products made or sold by Juniper Networks or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

#### *Network Configuration Example Validating a BGP-Based VPLS Multihoming Configuration*

Release 11.2

Copyright © 2011, Juniper Networks, Inc.

All rights reserved.

#### Revision History

April 2011—R1 Junos OS 11.2

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. The 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

**READ THIS END USER LICENSE AGREEMENT ("AGREEMENT") BEFORE DOWNLOADING, INSTALLING, OR USING THE SOFTWARE.** BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE OR OTHERWISE EXPRESSING YOUR AGREEMENT TO THE TERMS CONTAINED HEREIN, YOU (AS CUSTOMER OR IF YOU ARE NOT THE CUSTOMER, AS A REPRESENTATIVE/AGENT AUTHORIZED TO BIND THE CUSTOMER) CONSENT TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT OR CANNOT AGREE TO THE TERMS CONTAINED HEREIN, THEN (A) DO NOT DOWNLOAD, INSTALL, OR USE THE SOFTWARE, AND (B) YOU MAY CONTACT JUNIPER NETWORKS REGARDING LICENSE TERMS.

1. **The Parties.** The parties to this Agreement are (i) Juniper Networks, Inc. (if the Customer's principal office is located in the Americas) or Juniper Networks (Cayman) Limited (if the Customer's principal office is located outside the Americas) (such applicable entity being referred to herein as "Juniper"), and (ii) the person or organization that originally purchased from Juniper or an authorized Juniper reseller the applicable license(s) for use of the Software ("Customer") (collectively, the "Parties").

2. **The Software.** In this Agreement, "Software" means the program modules and features of the Juniper or Juniper-supplied software, for which Customer has paid the applicable license or support fees to Juniper or an authorized Juniper reseller, or which was embedded by Juniper in equipment which Customer purchased from Juniper or an authorized Juniper reseller. "Software" also includes updates, upgrades and new releases of such software. "Embedded Software" means Software which Juniper has embedded in or loaded onto the Juniper equipment and any updates, upgrades, additions or replacements which are subsequently embedded in or loaded onto the equipment.

3. **License Grant.** Subject to payment of the applicable fees and the limitations and restrictions set forth herein, Juniper grants to Customer a non-exclusive and non-transferable license, without right to sublicense, to use the Software, in executable form only, subject to the following use restrictions:

- a. Customer shall use Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.
- b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius or IMS AAA software on multiple computers or virtual machines (e.g., Solaris zones) requires multiple licenses, regardless of whether such computers or virtualizations are physically contained on a single chassis.
- c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface, processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.
- d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.
- e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

The foregoing license is not transferable or assignable by Customer. No license is granted herein to any user who did not originally purchase the applicable license(s) for the Software from Juniper or an authorized Juniper reseller.

4. **Use Prohibitions.** Notwithstanding the foregoing, the license provided herein does not permit the Customer to, and Customer agrees not to and shall not: (a) modify, unbundle, reverse engineer, or create derivative works based on the Software; (b) make unauthorized copies of the Software (except as necessary for backup purposes); (c) rent, sell, transfer, or grant any rights in and to any copy of the Software, in any form, to any third party; (d) remove any proprietary notices, labels, or marks on or in any copy of the Software or any product in which the Software is embedded; (e) distribute any copy of the Software to any third party, including as may be embedded in Juniper equipment sold in the secondhand market; (f) use any 'locked' or key-restricted feature, function, service, application, operation, or capability without first purchasing the applicable license(s) and obtaining a valid key from Juniper, even if such feature, function, service, application, operation, or capability is enabled without a key; (g) distribute any key for the Software provided by Juniper to any third party; (h) use the

Software in any manner that extends or is broader than the uses purchased by Customer from Juniper or an authorized Juniper reseller; (i) use Embedded Software on non-Juniper equipment; (j) use Embedded Software (or make it available for use) on Juniper equipment that the Customer did not originally purchase from Juniper or an authorized Juniper reseller; (k) disclose the results of testing or benchmarking of the Software to any third party without the prior written consent of Juniper; or (l) use the Software in any manner other than as expressly provided herein.

5. **Audit.** Customer shall maintain accurate records as necessary to verify compliance with this Agreement. Upon request by Juniper, Customer shall furnish such records to Juniper and certify its compliance with this Agreement.

6. **Confidentiality.** The Parties agree that aspects of the Software and associated documentation are the confidential property of Juniper. As such, Customer shall exercise all reasonable commercial efforts to maintain the Software and associated documentation in confidence, which at a minimum includes restricting access to the Software to Customer employees and contractors having a need to use the Software for Customer's internal business purposes.

7. **Ownership.** Juniper and Juniper's licensors, respectively, retain ownership of all right, title, and interest (including copyright) in and to the Software, associated documentation, and all copies of the Software. Nothing in this Agreement constitutes a transfer or conveyance of any right, title, or interest in the Software or associated documentation, or a sale of the Software, associated documentation, or copies of the Software.

8. **Warranty, Limitation of Liability, Disclaimer of Warranty.** The warranty applicable to the Software shall be as set forth in the warranty statement that accompanies the Software (the "Warranty Statement"). Nothing in this Agreement shall give rise to any obligation to support the Software. Support services may be purchased separately. Any such support shall be governed by a separate, written support services agreement. TO THE MAXIMUM EXTENT PERMITTED BY LAW, JUNIPER SHALL NOT BE LIABLE FOR ANY LOST PROFITS, LOSS OF DATA, OR COSTS OR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, THE SOFTWARE, OR ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. IN NO EVENT SHALL JUNIPER BE LIABLE FOR DAMAGES ARISING FROM UNAUTHORIZED OR IMPROPER USE OF ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. EXCEPT AS EXPRESSLY PROVIDED IN THE WARRANTY STATEMENT TO THE EXTENT PERMITTED BY LAW, JUNIPER DISCLAIMS ANY AND ALL WARRANTIES IN AND TO THE SOFTWARE (WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE), INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT DOES JUNIPER WARRANT THAT THE SOFTWARE, OR ANY EQUIPMENT OR NETWORK RUNNING THE SOFTWARE, WILL OPERATE WITHOUT ERROR OR INTERRUPTION, OR WILL BE FREE OF VULNERABILITY TO INTRUSION OR ATTACK. In no event shall Juniper's or its suppliers' or licensors' liability to Customer, whether in contract, tort (including negligence), breach of warranty, or otherwise, exceed the price paid by Customer for the Software that gave rise to the claim, or if the Software is embedded in another Juniper product, the price paid by Customer for such other product. Customer acknowledges and agrees that Juniper has set its prices and entered into this Agreement in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the Parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the Parties.

9. **Termination.** Any breach of this Agreement or failure by Customer to pay any applicable fees due shall result in automatic termination of the license granted herein. Upon such termination, Customer shall destroy or return to Juniper all copies of the Software and related documentation in Customer's possession or control.

10. **Taxes.** All license fees payable under this agreement are exclusive of tax. Customer shall be responsible for paying Taxes arising from the purchase of the license, or importation or use of the Software. If applicable, valid exemption documentation for each taxing jurisdiction shall be provided to Juniper prior to invoicing, and Customer shall promptly notify Juniper if their exemption is revoked or modified. All payments made by Customer shall be net of any applicable withholding tax. Customer will provide reasonable assistance to Juniper in connection with such withholding taxes by promptly: providing Juniper with valid tax receipts and other required documentation showing Customer's payment of any withholding taxes; completing appropriate applications that would reduce the amount of withholding tax to be paid; and notifying and assisting Juniper in any audit or tax proceeding related to transactions hereunder. Customer shall comply with all applicable tax laws and regulations, and Customer will promptly pay or reimburse Juniper for all costs and damages related to any liability incurred by Juniper as a result of Customer's non-compliance or delay with its responsibilities herein. Customer's obligations under this Section shall survive termination or expiration of this Agreement.

11. **Export.** Customer agrees to comply with all applicable export laws and restrictions and regulations of any United States and any applicable foreign agency or authority, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. Customer shall be liable for any such violations. The version of the Software supplied to Customer may contain encryption or other capabilities restricting Customer's ability to export the Software without an export license.

12. **Commercial Computer Software.** The Software is "commercial computer software" and is provided with restricted rights. Use, duplication, or disclosure by the United States government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7201 through 227.7202-4, FAR 12.212, FAR 27.405(b)(2), FAR 52.227-19, or FAR 52.227-14 (ALT III) as applicable.

13. **Interface Information.** To the extent required by applicable law, and at Customer's written request, Juniper shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of applicable fee, if any. Customer shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which Juniper makes such information available.

14. **Third Party Software.** Any licensor of Juniper whose software is embedded in the Software and any supplier of Juniper whose products or technology are embedded in (or services are accessed by) the Software shall be a third party beneficiary with respect to this Agreement, and such licensor or vendor shall have the right to enforce this Agreement in its own name as if it were Juniper. In addition, certain third party software may be provided with the Software and is subject to the accompanying license(s), if any, of its respective owner(s). To the extent portions of the Software are distributed under and subject to open source licenses obligating Juniper to make the source code for such portions publicly available (such as the GNU General Public License ("GPL") or the GNU Library General Public License ("LGPL")), Juniper will make such source code portions (including Juniper modifications, as appropriate) available upon request for a period of up to three years from the date of distribution. Such request can be made in writing to Juniper Networks, Inc., 1194 N. Mathilda Ave., Sunnyvale, CA 94089, ATTN: General Counsel. You may obtain a copy of the GPL at <http://www.gnu.org/licenses/gpl.html>, and a copy of the LGPL at <http://www.gnu.org/licenses/lgpl.html>.

15. **Miscellaneous.** This Agreement shall be governed by the laws of the State of California without reference to its conflicts of laws principles. The provisions of the U.N. Convention for the International Sale of Goods shall not apply to this Agreement. For any disputes arising under this Agreement, the Parties hereby consent to the personal and exclusive jurisdiction of, and venue in, the state and federal courts within Santa Clara County, California. This Agreement constitutes the entire and sole agreement between Juniper and the Customer with respect to the Software, and supersedes all prior and contemporaneous agreements relating to the Software, whether oral or written (including any inconsistent terms contained in a purchase order), except that the terms of a separate written agreement executed by an authorized Juniper representative and Customer shall govern to the extent such terms are inconsistent or conflict with terms contained herein. No modification to this Agreement nor any waiver of any rights hereunder shall be effective unless expressly assented to in writing by the party to be charged. If any portion of this Agreement is held invalid, the Parties agree that such invalidity shall not affect the validity of the remainder of this Agreement. This Agreement and associated documentation has been written in the English language, and the Parties agree that the English version will govern. (For Canada: Les parties aux présentes confirment leur volonté que cette convention de même que tous les documents y compris tout avis qui s'y rattache, soient rédigés en langue anglaise. (Translation: The parties confirm that this Agreement and all related documentation is and will be in the English language)).



# Table of Contents

Introduction .....	1
Understanding the Virtual Private LAN Service .....	3
Example: Configuring a Multihomed VPLS .....	5
SNMP MIB Support for BGP-Based VPLS .....	21
jnxVpnTable .....	21
jnxVpnIfTable .....	22
jnxVpnPwTable .....	23
jnxVpnRTTable .....	25
Enterprise-Specific VPN Traps .....	26
Verifying a Multihomed VPLS Connection Using SNMP MIB Walks .....	27



## 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 on page 5
- Overview on page 5
- Configuration on page 7
- Verification on page 18

### 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

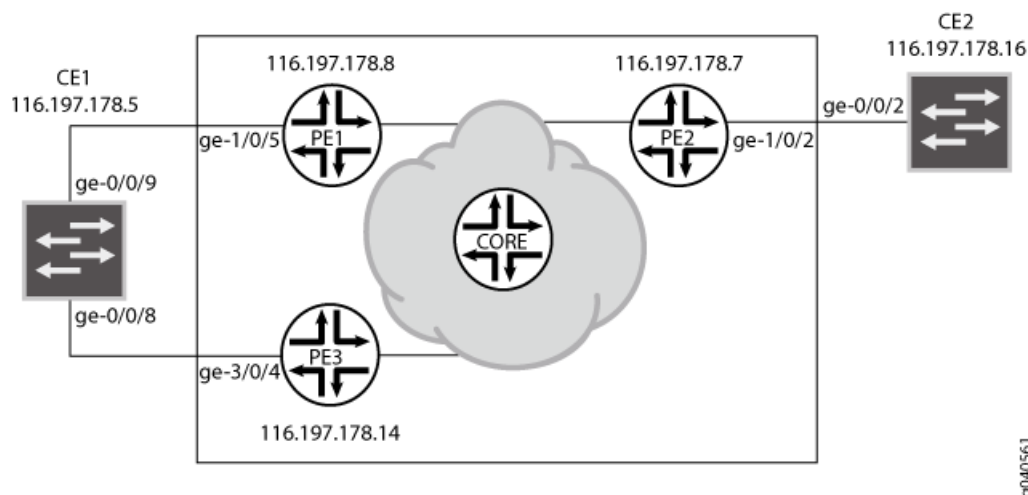
### Overview

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 on page 7
- Configuring Customer Edge Device CE2 on page 8
- Configuring Provider Edge Router PE1 on page 8
- Configuring Provider Edge Router PE2 on page 12
- Configuring Provider Edge Router PE3 on page 15

### 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)

Legend for connector 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 available
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](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** on page 21
- **jnxVpnIfTable** on page 22
- **jnxVpnPwTable** on page 23
- **jnxVpnRTTable** on page 25
- Enterprise-Specific VPN Traps on page 26

### 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
<b>jnxVpnRowStatus</b>	Creates, modifies, or deletes a row in this table.
<b>jnxVpnStorageType</b>	The storage type.
<b>jnxVpnDescription</b>	VPN description.
<b>jnxVpnIdentifierType</b>	Type of <b>jnxVpnIdentifier</b> .
<b>jnxVpnIdentifier</b>	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.
<b>jnxVpnConfiguredSites</b>	The number of sites configured in the VPN.
<b>jnxVpnActiveSites</b>	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
<b>jnxVpnIfProtocol</b>	<p>Indicates the protocol running over a VPN interface.</p> <p>This object contains the following values:</p> <ul style="list-style-type: none"> <li>• other(0)</li> <li>• frameRelay(1)</li> <li>• atmAal5(2)</li> <li>• atmCell(3)</li> <li>• ethernetVlan(4)</li> <li>• ethernet(5)</li> <li>• ciscoHdlc(6)</li> <li>• ppp(7)</li> <li>• cem(8)</li> <li>• atmVcc(9)</li> <li>• atmVpc(10)</li> <li>• vpls(11)</li> <li>• ipInter- working(12)</li> <li>• snapInter- working(13)</li> <li>• static(20)</li> <li>• rip(21)</li> <li>• ospf(22)</li> <li>• bgp(23)</li> <li>• atmTrunkNNI (129)</li> <li>• atmTrunkUNI (130)</li> </ul>
<b>jnxVpnIfInBandwidth</b>	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.
<b>jnxVpnIfOutBandwidth</b>	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.
<b>jnxVpnIfStatus</b>	<p>Status of a monitored VPN interface.</p> <p>This object contains the following values:</p> <ul style="list-style-type: none"> <li>• unknown(0)</li> <li>• noLocal- Interface(1)</li> <li>• disabled(2)</li> <li>• encapsulation- Mismatch(3)</li> <li>• down(4)</li> <li>• up(5)</li> </ul>

## 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
<b>jnxVpnPwRowStatus</b>	Creates, modifies, and deletes a row in this table.
<b>jnxVpnPwStorageType</b>	The storage type.
<b>jnxVpnPwAssociatedInterface</b>	The VPN index of the interface associated with a pseudowire. If no interface is associated with a pseudowire, 0 is returned.
<b>jnxVpnPwLocalSiteId</b>	The local site identifier for a pseudowire. When there is no local site identifier, 0 is returned.
<b>jnxVpnPwRemoteSiteId</b>	The remote site identifier. For example, the site at the end of the pseudowire. When there is no remote site identifier, 0 is returned.
<b>jnxVpnRemotePeldAddrType</b>	The remote PE address. For example, the router at the end of the pseudowire.
<b>jnxVpnRemotePeldAddress</b>	<p>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.</p> <p>This object can contain the following values:</p> <ul style="list-style-type: none"> <li>• <b>static(1)</b></li> <li>• <b>gre(2)</b></li> <li>• <b>l2tpv3(3)</b></li> <li>• <b>ipSec(4)</b></li> <li>• <b>ldp(5)</b></li> <li>• <b>rsvpTe(6)</b></li> <li>• <b>crLdp(7)</b></li> </ul>
<b>jnxVpnPwTunnelType</b>	The type of tunnel over which the pseudowire is carried.
<b>jnxVpnPwTunnelName</b>	The name of the tunnel over which the pseudowire is carried.
<b>jnxVpnPwReceiveDemux</b>	The demultiplexer value that identifies received packets associated with this pseudowire.
<b>jnxVpnPwTransmitDemux</b>	The demultiplexer value that identifies the transmitted packets associated with this pseudowire.

Table 3: jnxVpnPwTable Objects (*continued*)

Object	Description
jnxVpnPwStatus	<p>The status of the pseudowire.</p> <p>This object can have the following values:</p> <ul style="list-style-type: none"> <li>• <b>unknown(0)</b></li> <li>• <b>down(1)</b></li> <li>• <b>up(2)</b></li> </ul>
jnxVpnPwTunnelStatus	The status of the PE-to-PE tunnel over which the pseudowire is carried.
jnxVpnPwRemoteSiteStatus	<p>The interface status at the remote end of the pseudowire.</p> <p>This object can have the following values:</p> <ul style="list-style-type: none"> <li>• <b>unknown(0)</b></li> <li>• <b>outOfRange(1)</b></li> <li>• <b>down(2)</b></li> <li>• <b>up(3)</b></li> </ul>
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
<b>jnxVpnRTRowStatus</b>	Creates, modifies, or deletes a row in this table.
<b>jnxVpnRTStorageType</b>	Identifies the storage type for an object.
<b>jnxVpnRTType</b>	The type of the following route target. The type can be <b>routeTarget[012]</b> or <b>none</b> .
<b>jnxVpnRT</b>	The VPN route target. If <b>jnxVpnRTType</b> is <b>none</b> , the value must be all zeros.
<b>jnxVpnRTFunction</b>	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
<b>jnxVpnIfUp</b>	Indicates that the interface with the index <b>jnxVpnIfIndex</b> belonging to the <b>jnxVpnIfVpnName</b> of type <b>jnxVpnIfVpnType</b> came up.
<b>jnxVpnIfDown</b>	Indicates that the interface with index <b>jnxVpnIfIndex</b> belonging to <b>jnxVpnIfVpnName</b> of type <b>jnxVpnIfVpnType</b> went down.
<b>jnxVpnPwUp</b>	Indicates that the pseudowire with the index <b>jnxVpnPwIndex</b> belonging to <b>jnxVpnPwVpnName</b> of type <b>jnxVpnPwVpnType</b> came up.
<b>jnxVpnPwDown</b>	Indicates that the pseudowire with index <b>jnxVpnPwIndex</b> belonging to <b>jnxVpnPwVpnName</b> of type <b>jnxVpnPwVpnType</b> 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](#)

