Configuring Virtual Private LAN Service (VPLS) and VPLS BGP-Based Autodiscovery

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Configuring VPLS

Information About VPLS

VPLS Overview

VPLS (Virtual Private LAN Service) enables enterprises to link together their Ethernet-based LANs from multiple sites via the infrastructure provided by their service provider. From the enterprise perspective, the service provider's public network looks like one giant Ethernet LAN. For the service provider, VPLS provides an opportunity to deploy another revenue-generating service on top of their existing network without major capital expenditures. Operators can extend the operational life of equipment in their network.

Virtual Private LAN Services (VPLS) uses the provider core to join multiple attachment circuits together to simulate a virtual bridge that connects the multiple attachment circuits together. From a customer point of view, there is no topology for VPLS. All of the CE devices appear to connect to a logical bridge emulated by the provider core.
Full-Mesh Configuration

The full-mesh configuration requires a full mesh of tunnel label switched paths (LSPs) between all the PEs that participate in the VPLS. With full-mesh, signaling overhead and packet replication requirements for each provisioned VC on a PE can be high.

You set up a VPLS by first creating a virtual forwarding instance (VFI) on each participating PE router. The VFI specifies the VPN ID of a VPLS domain, the addresses of other PE devices in the domain, and the type of tunnel signaling and encapsulation mechanism for each peer PE router.

The set of VFIs formed by the interconnection of the emulated VCs is called a VPLS instance; it is the VPLS instance that forms the logic bridge over a packet switched network. The VPLS instance is assigned a unique VPN ID.

The PE devices use the VFI to establish a full-mesh LSP of emulated VCs to all the other PE devices in the VPLS instance. PE devices obtain the membership of a VPLS instance through static configuration using the Cisco IOS CLI.

The full-mesh configuration allows the PE router to maintain a single broadcast domain. Thus, when the PE router receives a broadcast, multicast, or unknown unicast packet on an attachment circuit, it sends the packet out on all other attachment circuits and emulated circuits to all other CE devices participating in that VPLS instance. The CE devices see the VPLS instance as an emulated LAN.

To avoid the problem of a packet looping in the provider core, the PE devices enforce a "split-horizon" principle for the emulated VCs. That means if a packet is received on an emulated VC, it is not forwarded on any other emulated VC.

After the VFI has been defined, it needs to be bound to an attachment circuit to the CE device.

The packet forwarding decision is made by looking up the Layer 2 virtual forwarding instance (VFI) of a particular VPLS domain.

A VPLS instance on a particular PE router receives Ethernet frames that enter on specific physical or logical ports and populates a MAC table similarly to how an Ethernet switch works. The PE router can use the MAC address to switch those frames into the appropriate LSP for delivery to the another PE router at a remote site.

If the MAC address is not in the MAC address table, the PE router replicates the Ethernet frame and floods it to all logical ports associated with that VPLS instance, except the ingress port where it just entered. The PE router updates the MAC table as it receives packets on specific ports and removes addresses not used for specific periods.
VPLS BGP Based Autodiscovery

VPLS Autodiscovery enables each Virtual Private LAN Service (VPLS) provider edge (PE) device to discover other PE devices that are part of the same VPLS domain. VPLS Autodiscovery also tracks PE devices when they are added to or removed from a VPLS domain. As a result, with VPLS Autodiscovery enabled, you no longer need to manually configure a VPLS domain and maintain the configuration when a PE device is added or deleted. VPLS Autodiscovery uses the Border Gateway Protocol (BGP) to discover VPLS members and set up and tear down pseudowires in a VPLS domain.

BGP uses the Layer 2 VPN (L2VPN) Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 virtual forwarding instance (VFI) is configured. The prefix and path information is stored in the L2VPN database, which allows BGP to make decisions about the best path. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, this endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

The BGP autodiscovery mechanism facilitates the configuration of L2VPN services, which are an integral part of the VPLS feature. VPLS enables flexibility in deploying services by connecting geographically dispersed sites as a large LAN over high-speed Ethernet in a robust and scalable IP Multiprotocol Label Switching (MPLS) network.

For scale information related to this feature, see Cisco Catalyst 9300 Series Switches Data Sheet.

Restrictions for VPLS

- Protocol-based CLI Method (interface pseudowire configuration) is not supported. Only VFI and Xconnect mode are supported.
- Flow-Aware Transport Pseudowire (FAT PW) is not supported.
- IGMP Snooping is not Supported. Multicast traffic floods with IGMP Snooping disabled.
- L2 Protocol Tunneling is not supported.
- Integrated Routing and Bridging (IRB) not supported.
- Virtual Circuit Connectivity Verification (VCCV) ping with explicit null is not supported.
- The switch is supported only as spoke in H-VPLS but not as hub.
- L2 VPN Interworking is not supported.
- `ip unnumbered` command is not supported in MPLS configuration.
- VC statistics are not displayed for flood traffic in the output of `show mpls l2 vc vcid detail` command.
- dot1q tunnel is not supported in the attachment circuit.

Configuring PE Layer 2 Interfaces to CEs

Configuring 802.1Q Trunks for Tagged Traffic from a CE

**SUMMARY STEPS**

1. `enable`
2. `configure terminal`
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 1.   | enable            | Enables privileged EXEC mode.  
|      | Example:          | • Enter your password if prompted.  
|      | Device> enable    |         |
| 2.   | configure terminal| Enters global configuration mode.  
|      | Example:          |         |
|      | Device# configure terminal | |
| 3.   | interface interface-id | Defines the interface to be configured as a trunk, and enters interface configuration mode.  
|      | Example:          |         |
|      | Device(config)# interface TenGigabitEthernet1/0/24 | |
| 4.   | no ip address ip_address mask [secondary] | Disables IP processing and enters interface configuration mode.  
|      | Example:          |         |
|      | Device(config-if)# no ip address | |
| 5.   | switchport       | Modifies the switching characteristics of the Layer 2-switched interface.  
|      | Example:          |         |
|      | Device(config-if)# switchport | |
| 6.   | switchport trunk encapsulation dot1q | Sets the switch port encapsulation format to 802.1Q.  
|      | Example:          |         |
|      | Device(config-if)# switchport trunk encapsulation dot1q | |
| 7.   | switchport trunk allow vlan vlan_ID | Sets the list of allowed VLANs.  
|      | Example:          |         |

**Configuring 802.1Q Trunks for Tagged Traffic from a CE**
### Command or Action

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Command or Action</th>
</tr>
</thead>
</table>
| Sets the interface to a trunking VLAN Layer 2 interface. | `Device(config-if)# switchport trunk allow vlan 2129`
| Returns to privileged EXEC mode. | `Device(config-if)# switchport mode trunk`

### Step 8

#### Example:

Device(config-if)# switchport mode trunk

### Step 9

#### Example:

Device(config)# end

## Configuring 802.1Q Access Ports for Untagged Traffic from a CE

### SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `interface interface-id`
4. `no ip address ip_address mask [secondary ]`
5. `switchport`
6. `switchport mode access`
7. `switchport access vlan vlan_ID`
8. `end`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Enables privileged EXEC mode. | `enable`
| Enter your password if prompted. | `Device> enable`
| Enters global configuration mode. | `configure terminal`
| | `Device# configure terminal`
| Defines the interface to be configured as a trunk, and enters interface configuration mode. | `interface interface-id`
| | `Device# interface interface-id`
### Configuring Layer 2 VLAN Instances on a PE

Configuring the Layer 2 VLAN interface on the PE enables the Layer 2 VLAN instance on the PE router to the VLAN database to set up the mapping between the VPLS and VLANs.

#### SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `vlan vlan-id`
4. `interface vlan vlan-id`
5. `end`
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>- Enter your password if prompted.</td>
</tr>
<tr>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> vlan vlan-id</td>
<td>Configures a specific virtual LAN (VLAN).</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config)# vlan 2129</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> interface vlan vlan-id</td>
<td>Configures an interface on the VLAN.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config-vlan)# interface vlan 2129</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> end</td>
<td>Returns to privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config)# end</td>
<td></td>
</tr>
</tbody>
</table>

**Configuring MPLS in the PE**

To configure MPLS in the PE, you must provide the required MPLS parameters.

**SUMMARY STEPS**

1. enable
2. configure terminal
3. mpls ip
4. mpls label protocol ldp
5. end
6. mpls ldp logging neighbor-changes

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
</tbody>
</table>

**Configuring Virtual Private LAN Service (VPLS) and VPLS BGP-Based Autodiscovery**
### Configuring VFI in the PE

The virtual switch instance (VFI) specifies the VPN ID of a VPLS domain, the addresses of other PE devices in this domain, and the type of tunnel signaling and encapsulation mechanism for each peer (This is where you create the VFI and associated VCs.). Configure a VFI as follows:

**SUMMARY STEPS**

1. `enable`
2. `configure terminal`
3. `l2 vfi vfi-name manual`
4. `vpn id vpn-id`
5. `neighbor remote-router-id {encapsulation mpls}`
6. `end`

---

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Device&gt; <em>enable</em></td>
<td>• Enter your password if prompted.</td>
</tr>
</tbody>
</table>

**Step 2**

**configure terminal**

**Example:**

Device# **configure terminal**

**Step 3**

**mpls ip**

**Example:**

Device(config)# **mpls ip**

**Step 4**

**mpls label protocol ldp**

**Example:**

Device(config-vlan)# **mpls label protocol ldp**

**Step 5**

**end**

**Example:**

Device(config)# **end**

**Step 6**

**mpls ldp logging neighbor-changes**

**Example:**

Device(config)# **mpls ldp logging neighbor-changes**

(Optional) Determines logging neighbor changes.
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Step 1 | enable | Enables privileged EXEC mode.  
**Example:**  
Device> enable  
- Enter your password if prompted. |
| Step 2 | configure terminal | Enters global configuration mode.  
**Example:**  
Device# configure terminal |
| Step 3 | l2 vfi vfi-name manual | Enables the Layer 2 VFI manual configuration mode.  
**Example:**  
Device(config)# l2 vfi 2129 manual |
| Step 4 | vpn id vpn-id | Configures a VPN ID for a VPLS domain. The emulated VCs bound to this Layer 2 VRF use this VPN ID for signaling.  
**Note**  
vpn-id is the same as vlan-id.  
**Example:**  
Device(config-vfi)# vpn id 2129 |
| Step 5 | neighbor remote-router-id {encapsulation mpls} | Specifies the remote peering router ID and the tunnel encapsulation type or the pseudo-wire property to be used to set up the emulated VC.  
**Example:**  
Device(config-vfi)# neighbor remote-router-id {encapsulation mpls} |
| Step 6 | end | Returns to privileged EXEC mode.  
**Example:**  
Device(config)# end |

## Associating the Attachment Circuit with the VFI at the PE

After defining the VFI, you must bind it to one or more attachment circuits.

## SUMMARY STEPS

1. enable  
2. configure terminal  
3. interface vlan vlan-id  
4. no ip address  
5. xconnect vfi vfi-name
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 1    | `enable`          | Enables privileged EXEC mode.  
*Example:*  
Device> `enable`  
| 2    | `configure terminal` | Enters global configuration mode.  
*Example:*  
Device# `configure terminal`  
| 3    | `interface vlan vlan-id` | Creates or accesses a dynamic switched virtual interface (SVI).  
*Note*  
*vlan-id* is the same as *vpn-id*.  
*Example:*  
Device(config)# `interface vlan 2129`  
| 4    | `no ip address`   | Disables IP processing. (You configure a Layer 3 interface for the VLAN if you configure an IP address.)  
*Example:*  
Device(config-if)# `no ip address`  
| 5    | `xconnect vfi vfi-name` | Specifies the Layer 2 VFI that you are binding to the VLAN port.  
*Example:*  
Device(config-if)# `xconnect vfi 2129`  
| 6    | `end`             | Returns to privileged EXEC mode.  
*Example:*  
Device(config)# `end`
Configuration Examples for VPLS

The `show mpls 12transport vc detail` command provides information the virtual circuits.

Local interface: VFI 2129 vfi up
  Interworking type is Ethernet
  Destination address: 44.254.44.44, VC ID: 2129, VC status: up
  Output interface: Gi1/0/9, imposed label stack (18 17)
  Preferred path: not configured
  Default path: active
  Next hop: 177.77.177.2
  Create time: 19:09:33, last status change time: 09:24:14
  Last label FSM state change time: 09:24:14
  Signaling protocol: LDP, peer 44.254.44.44:0 up
  Targeted Hello: 1.1.1.72(LDP Id) -> 44.254.44.44, LDP is UP
  Graceful restart: configured and enabled
  Non stop routing: not configured and not enabled
  Status TLV support (local/remote) : enabled/supported
  LDP route watch : enabled
Label/status state machine : established, LruRru
Last local dataplane status rcvd: No fault
Last BFD dataplane status rcvd: Not sent
Last BFD peer monitor status rcvd: No fault
Last local AC circuit status rcvd: No fault
Last local AC circuit status sent: No fault
Last local PW i/f circ status rcvd: No fault
Last local LDP TLV status rcvd: No fault
Last remote LDP TLV status rcvd: No fault
Last remote LDP ADJ status rcvd: No fault
MPLS VC labels: local 512, remote 17
  Group ID: local n/a, remote 0
  MTU: local 1500, remote 1500
Remote interface description:
  Sequencing: receive disabled, send disabled
  Control Word: Off
SSO Descriptor: 44.254.44.44/2129, local label: 512
Dataplane:
  SSM segment/switch IDs: 20498/20492 (used), PWID: 2
VC statistics:
  transit packet totals: receive 0, send 0
  transit byte totals: receive 0, send 0
  transit packet drops: receive 0, seq error 0, send 0

The show l2vpn atom vc shows that ATM over MPLS is configured on a VC.

pseudowire100005 is up, VC status is up PW type: Ethernet
Create time: 19:25:56, last status change time: 09:40:37
Last label FSM state change time: 09:40:37
Destination address: 44.254.44.44 VC ID: 2129
  Output interface: Gi1/0/9, imposed label stack {18 17}
  Preferred path: not configured
  Default path: active
  Next hop: 177.77.177.2
Member of vfi service 2129
  Bridge-Domain id: 2129
  Service id: 0x32000003
Signaling protocol: LDP, peer 44.254.44.44:0 up
  Targeted Hello: 1.1.1.72(LDP Id) -> 44.254.44.44, LDP is UP
Graceful restart: configured and enabled
Non stop routing: not configured and not enabled
PWid FEC (128), VC ID: 2129
Status TLV support (local/remote) : enabled/supported
  LDP route watch : enabled
  Label/status state machine : established, LruRru
  Local dataplane status received : No fault
  BFD dataplane status received : Not sent
  BFD peer monitor status received : No fault
  Status received from access circuit : No fault
  Status sent to access circuit : No fault
  Status received from pseudowire i/f : No fault
Configuring VPLS BGP-based Autodiscovery

Information About VPLS BGP-Based Autodiscovery

VPLS BGP Based Autodiscovery

VPLS Autodiscovery enables each Virtual Private LAN Service (VPLS) provider edge (PE) device to discover other PE devices that are part of the same VPLS domain. VPLS Autodiscovery also tracks PE devices when they are added to or removed from a VPLS domain. As a result, with VPLS Autodiscovery enabled, you no longer need to manually configure a VPLS domain and maintain the configuration when a PE device is added or deleted. VPLS Autodiscovery uses the Border Gateway Protocol (BGP) to discover VPLS members and set up and tear down pseudowires in a VPLS domain.

BGP uses the Layer 2 VPN (L2VPN) Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 virtual forwarding instance (VFI) is configured. The prefix and path information is stored in the L2VPN database, which allows BGP to make decisions about the best path. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, this endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.
The BGP autodiscovery mechanism facilitates the configuration of L2VPN services, which are an integral part of the VPLS feature. VPLS enables flexibility in deploying services by connecting geographically dispersed sites as a large LAN over high-speed Ethernet in a robust and scalable IP Multiprotocol Label Switching (MPLS) network.

For scale information related to this feature, see Cisco Catalyst 9300 Series Switches Data Sheet.

Enabling VPLS BGP-based Autodiscovery

Perform this task to enable Virtual Private LAN Service (VPLS) PE devices to discover other PE devices that are part of the same VPLS domain.

SUMMARY STEPS

1. enable
2. configure terminal
3. l2 vfi vfi-name autodiscovery
4. vpn id vpn-id
5. end

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1** enable | Enables privileged EXEC mode.  
  Example:  
  Device> enable |
| **Step 2** configure terminal | Enters global configuration mode.  
  Example:  
  Device# configure terminal |
| **Step 3** l2 vfi vfi-name autodiscovery | Enables VPLS Autodiscovery on a PE device and enters L2 VFI configuration mode.  
  Example:  
  Device(config)# l2 vfi 2128 autodiscovery |
| **Step 4** vpn id vpn-id | Configures a VPN ID for the VPLS domain.  
  Example:  
  Device(config-vfi)# vpn id 2128 |
| **Step 5** end | Returns to privileged EXEC mode.  
  Example:  
  Device(config)# end |
Configuring BGP to Enable VPLS Autodiscovery

The Border Gateway Protocol (BGP) Layer 2 VPN (L2VPN) address family supports a separate L2VPN Routing Information Base (RIB) that contains endpoint provisioning information for Virtual Private LAN Service (VPLS) Autodiscovery. BGP learns the endpoint provisioning information from the L2VPN database, which is updated each time a Layer 2 virtual forwarding instance (VFI) is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

SUMMARY STEPS

1. enable
2. configure terminal
3. router bgp autonomous-system-number
4. no bgp default ipv4-unicast
5. bgp log-neighbor-changes
6. neighbor remote-as { ip-address | peer-group-name } remote-as autonomous-system-number
7. neighbor { ip-address | peer-group-name } update-source interface-type interface-number
8. Repeat Steps 6 and 7 to configure other BGP neighbors.
9. address-family l2vpn [vpls]
10. neighbor { ip-address | peer-group-name } activate
11. neighbor { ip-address | peer-group-name } send-community { both | standard | extended }
12. Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.
13. exit-address-family
14. end

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>enable</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Device&gt; enable</strong></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>configure terminal</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><strong>Device# configure terminal</strong></td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
</tr>
</tbody>
</table>
| 3    | `router bgp autonomous-system-number`  
**Example:**  
Device(config)# `router bgp 1000` | Enters router configuration mode for the specified routing process. |
| 4    | `no bgp default ipv4-unicast`  
**Example:**  
Device(config-router)# `no bgp default ipv4-unicast` | Disables the IPv4 unicast address family for the BGP routing process.  
**Note:** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured using the neighbor remote-as router configuration command unless you configure the `no bgp default ipv4-unicast` router configuration command before configuring the neighbor remote-as command. Existing neighbor configurations are not affected. |
| 5    | `bgp log-neighbor-changes`  
**Example:**  
Device(config-router)# `bgp log-neighbor-changes` | Enables logging of BGP neighbor resets. |
| 6    | `neighbor remote-as { ip-address | peer-group-name } remote-as autonomous-system-number`  
**Example:**  
Device(config-router)# `neighbor 44.254.44.44 remote-as 1000` | Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.  
- If the autonomous-system-number argument matches the autonomous system number specified in the router `bgp` command, the neighbor is an internal neighbor.  
- If the autonomous-system-number argument does not match the autonomous system number specified in the router `bgp` command, the neighbor is an external neighbor. |
| 7    | `neighbor { ip-address | peer-group-name } update-source interface-type interface-number`  
**Example:**  
Device(config-router)# `neighbor 44.254.44.44 update-source Loopback300` | (Optional) Configures a device to select a specific source or interface to receive routing table updates. |
| 8    | Repeat Steps 6 and 7 to configure other BGP neighbors. | Exits interface configuration mode. |
| 9    | `address-family l2vpn [vpls]`  
**Example:** | Specifies the L2VPN address family and enters address family configuration mode. |
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Device(config-router)# address-family l2vpn vpls</code></td>
<td>The optional <code>vpls</code> keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers.</td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td><strong>Purpose</strong></td>
</tr>
<tr>
<td>`neighbor { ip-address</td>
<td>peer-group-name } activate`</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>Device(config-router-af)# neighbor 44.254.44.44 activate</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 11</strong></td>
<td><strong>Purpose</strong></td>
</tr>
<tr>
<td>`neighbor { ip-address</td>
<td>peer-group-name } send-community { both</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>Device(config-router-af)# neighbor 44.254.44.44 send-community both</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 12</strong></td>
<td>Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.</td>
</tr>
<tr>
<td><strong>Step 13</strong></td>
<td><strong>Purpose</strong></td>
</tr>
<tr>
<td><code>exit-address-family</code></td>
<td>Exits address family configuration mode and returns to router configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>Device(config-router-af)# exit-address-family</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 14</strong></td>
<td><strong>Purpose</strong></td>
</tr>
<tr>
<td><code>end</code></td>
<td>Exits router configuration mode and returns to privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>Device(config-router-af)# end</code></td>
<td></td>
</tr>
</tbody>
</table>
Configuration Examples for VPLS BGP-AD

PE Configuration

```conf
router bgp 1000
bgp log-neighbor-changes
bgp graceful-restart
neighbor 44.254.44.44 remote-as 1000
neighbor 44.254.44.44 update-source Loopback300
!
address-family l2vpn vpls
neighbor 44.254.44.44 activate
neighbor 44.254.44.44 send-community both
exit-address-family
!
l2 vfi 2128 autodiscovery
vpn id 2128
interface Vlan2128
no ip address
xconnect vfi 2128
```

The following is a sample output of `show platform software fed sw 1 matm macTable vlan 2000` command:

<table>
<thead>
<tr>
<th>VLAN</th>
<th>MAC address</th>
<th>Type</th>
<th>Seq#</th>
<th>maHandle</th>
<th>siHandle</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>2852.6134.05c8</td>
<td>0X8002</td>
<td>0</td>
<td>0xffbba312c8</td>
<td>0xffb9ef938</td>
</tr>
<tr>
<td></td>
<td>0x5154</td>
<td>0</td>
<td>0</td>
<td>Vlan2000</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>0000.0078.9012</td>
<td>0X1</td>
<td>32627</td>
<td>0xffbb665ec8</td>
<td>0xffbb60b198</td>
</tr>
<tr>
<td></td>
<td>0xffbb653f98</td>
<td>300</td>
<td>27844</td>
<td>Port-channel11</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>2852.6134.0000</td>
<td>0X1</td>
<td>32651</td>
<td>0xffba5e1a8</td>
<td>0xff454c2328</td>
</tr>
<tr>
<td></td>
<td>0xffbb653f98</td>
<td>300</td>
<td>63</td>
<td>Port-channel11</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>0000.0012.3456</td>
<td>0X20000001</td>
<td>32655</td>
<td>0xffba5c508</td>
<td>0xff44f9ec98</td>
</tr>
<tr>
<td></td>
<td>0x0</td>
<td>300</td>
<td>1</td>
<td>2000:33.33.33.33</td>
<td></td>
</tr>
</tbody>
</table>

Total Mac number of addresses:: 4
*a_time=aging_time(secs)  *e_time=total_elapsed_time(secs)

Type:
- MAT_DYNAMIC_ADDR 0x1
- MAT_CPU_ADDR 0x4
- MAT_ALL_VLANS 0x10
- MAT_IPMULT_ADDR 0x40
- MAT_DO NOT AGE 0x100
- MAT_NO_PORT 0x400
- MAT_UP ADDR 0x1000
- MAT_DOT1X ADDR 0x4000
- MAT_WIRELESS_ADDR 0x10000
- MAT_OPQ_DATA_PRESENT 0x40000
- MAT_DL_ADDR 0x100000
- MAT_MSRP_ADDR 0x400000
- MAT_LISP_REMOTE_ADDR 0x1000000

The following is a sample output of `show bgp l2vpn vpls all` command:
BGP table version is 6, local router ID is 222.5.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
  r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
  x best-external, a additional-path, c RIB-compressed,
  t secondary path,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Metric</th>
<th>LocPrf</th>
<th>Weight</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>*&gt; 1000:2128:1.1.1.72/96</td>
<td>0.0.0.0</td>
<td>32768</td>
<td></td>
<td></td>
<td>?</td>
</tr>
<tr>
<td>*&gt;i 1000:2128:44.254.44.44/96</td>
<td>44.254.44.44</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>?</td>
</tr>
</tbody>
</table>