

VPLS Autodiscovery BGP Based

VPLS Autodiscovery enables Virtual Private LAN Service (VPLS) provider edge (PE) devices to discover other PE devices that are part of the same VPLS domain. VPLS Autodiscovery also automatically detects when PE devices 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.

This module describes how to configure BGP-based VPLS Autodiscovery.

- Restrictions for VPLS Autodiscovery BGP Based, on page 1
- Information About VPLS Autodiscovery BGP Based, on page 2
- How to Configure VPLS Autodiscovery BGP Based, on page 5
- Configuration Examples for VPLS Autodiscovery BGP Based, on page 24
- Additional References for VPLS Autodiscovery BGP Based, on page 32
- Feature Information for VPLS Autodiscovery BGP Based, on page 33

Restrictions for VPLS Autodiscovery BGP Based

- Virtual Private LAN Service (VPLS) Autodiscovery supports only IPv4 addresses.
- VPLS Autodiscovery uses Forwarding Equivalence Class (FEC) 129 to convey endpoint information. Manually configured pseudowires use FEC 128.
- VPLS Autodiscovery is not supported with Layer 2 Tunnel Protocol Version 3 (L2TPv3).
- You can configure both autodiscovered and manually configured pseudowires in a single virtual forwarding instance (VFI). However, you cannot configure different pseudowires on the same peer PE device.
- After enabling VPLS Autodiscovery, if you manually configure a neighbor by using the **neighbor** command and both peers are in autodiscovery mode, each peer will receive discovery data for that VPLS. To prevent peers from receiving data for the VPLS domain, manually configure route target (RT) values.
- If you manually configure multiple pseudowires and target different IP addresses on the same PE device for each pseudowire, do not use the same virtual circuit (VC) ID to identify pseudowires that terminate at the same PE device.
- If you manually configure a neighbor on one PE device, you cannot configure the same pseudowire in the other direction by using autodiscovery on another PE device.

- Tunnel selection is not supported with autodiscovered neighbors.
- Up to 16 RTs are supported per VFI.
- The same RT is not allowed in multiple VFIs on the same PE device.
- The Border Gateway Protocol (BGP) autodiscovery process does not support dynamic, hierarchical VPLS. User-facing PE (U-PE) devices cannot discover network-facing PE (N-PE) devices, and N-PE devices cannot discover U-PE devices.
- Pseudowires for autodiscovered neighbors have split horizon enabled. (A split horizon is enabled by default on all interfaces. A split horizon blocks route information from being advertised by a device, irrespective of the interface from which the information originates.) Therefore, manually configure pseudowires for hierarchical VPLS. Ensure that U-PE devices do not participate in BGP autodiscovery for these pseudowires.
- Do not disable split horizon on autodiscovered neighbors. Split horizon is required with VPLS Autodiscovery.
- The provisioned peer address must be a /32 address bound to the peer's Label Distribution Protocol (LDP) router ID.
- A peer PE device must be able to access the IP address that is used as the local LDP router ID. Even if the IP address is not used in the **xconnect** command on the peer PE device, the IP address must be reachable.

Information About VPLS Autodiscovery BGP Based

How VPLS Works

Virtual Private LAN Service (VPLS) allows Multiprotocol Label Switching (MPLS) networks to provide multipoint Ethernet LAN services, also known as Transparent LAN Services (TLS). All customer sites in a VPLS appear to be on the same LAN, even though these sites might be in different geographic locations.

How the VPLS Autodiscovery BGP Based Feature Works

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. Autodiscovery and signaling functions use the Border Gateway Protocol (BGP) to find and track PE devices.

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 more information about BGP and the L2VPN address family in relation to VPLS Autodiscovery, see the following chapters in the *IP Routing: BGP Configuration Guide*:

- "BGP Support for the L2VPN Address Family" chapter

How Enabling VPLS Autodiscovery Differs from Manually Configuring VPLS

With VPLS Autodiscovery enabled, you no longer need to manually set up Virtual Private LAN Service (VPLS). The commands that you use to set up VPLS Autodiscovery are similar to those that you use to manually configure VPLS, as shown in the table below. VPLS Autodiscovery uses **neighbor** commands in L2VPN address family mode to distribute endpoint information to configure a pseudowire.

Table 1: Manual VPLS Configuration Versus VPLS Autodiscovery Configuration

Manual Configuration of VPLS	VPLS Autodiscovery BGP Based

`l2 vfi vpls1 manual`
` vpn id 100`
` neighbor 10.10.10.1 encapsulation mpls`
` neighbor 10.10.10.0 encapsulation mpls`
` exit`
	`l2 vfi vpls1 autodiscovery`
	` vpn id 100`
	` exit`
	`router bgp 1`
	` no bgp default ipv4-unicast`
	` bgp log-neighbor-changes`
	` bgp update-delay 1`
	` neighbor 10.1.1.2 remote-as 1`
	` neighbor 10.1.1.2 update-source Loopback1`
	` .`
	` .`
	` .`
	` address-family l2vpn vpls`
	` neighbor 10.1.1.2 activate`
	` neighbor 10.1.1.2 send-community extended`
	`exit-address-family`

Configure VPLS Autodiscovery by using the **l2 vfi autodiscovery** command. This command allows a virtual forwarding instance (VFI) to learn and advertise pseudowire endpoints. As a result, you no longer need to enter the **neighbor** command in L2 VFI configuration mode.

However, the **neighbor** command is still supported with VPLS Autodiscovery in L2 VFI configuration mode. You can use the **neighbor** command to allow PE devices that do not participate in the autodiscovery process to join the VPLS domain. You can also use the **neighbor** command with PE devices that have been configured using the Tunnel Selection feature. In addition, you can use the **neighbor** command in hierarchical VPLS configurations that have user-facing PE (U-PE) devices that do not participate in the autodiscovery process and have split-horizon forwarding disabled.

How Enabling VPLS Autodiscovery Differs from Manually Configuring VPLS using the commands associated with the L2VPN Protocol-Based CLIs feature

With VPLS Autodiscovery enabled, you no longer need to manually set up Virtual Private LAN Service (VPLS). The commands that you use to set up VPLS Autodiscovery are similar to those that you use to manually configure VPLS, as shown in the table below. VPLS Autodiscovery uses **neighbor** commands in L2VPN address family mode to distribute endpoint information to configure a pseudowire.

Table 2: Manual VPLS Configuration Versus VPLS Autodiscovery Configuration

Manual Configuration of VPLS	VPLS Autodiscovery BGP Based

`l2vpn vfi context vpls1`
` vpn id 100`
` neighbor 10.10.10.1 encapsulation mpls`
` neighbor 10.10.10.0 encapsulation mpls`
` exit`
	`l2vpn vfi context vpls1`
	` vpn id 100`
	` autodiscovery bgp signaling ldp`
	` exit`
	`router bgp 1`
	` no bgp default ipv4-unicast`
	` bgp log-neighbor-changes`
	` bgp update-delay 1`
	` neighbor 10.1.1.2 remote-as 1`
	` neighbor 10.1.1.2 update-source Loopback1`
	` .`
	` .`
	` .`
	` address-family l2vpn vpls`
	` neighbor 10.1.1.2 activate`
	` neighbor 10.1.1.2 send-community extended`
	`exit-address-family`

Configure VPLS Autodiscovery by using the **autodiscovery** command. This command allows a virtual forwarding instance (VFI) to learn and advertise pseudowire endpoints. As a result, you no longer need to enter the **neighbor** command in L2 VFI configuration mode.

However, the **neighbor** command is still supported with VPLS Autodiscovery in L2 VFI configuration mode. You can use the **neighbor** command to allow PE devices that do not participate in the autodiscovery process to join the VPLS domain. You can also use the **neighbor** command with PE devices that have been configured using the Tunnel Selection feature. In addition, you can use the **neighbor** command in hierarchical VPLS configurations that have user-facing PE (U-PE) devices that do not participate in the autodiscovery process and have split-horizon forwarding disabled.

show Commands Affected by VPLS Autodiscovery BGP Based

The following **show** commands were enhanced for VPLS Autodiscovery:

- The **show mpls l2transport vc detail** command was updated to include Forwarding Equivalence Class (FEC) 129 signaling information for autodiscovered Virtual Private LAN Service (VPLS) pseudowires.
- The **show vfi** command was enhanced to display information related to autodiscovered virtual forwarding instances (VFIs). The new output includes the VPLS ID, the route distinguisher (RD), the route target (RT), and router IDs of discovered peers.
- The **show xconnect** command was updated with the **rib** keyword to provide Routing Information Base (RIB) information about pseudowires.

BGP VPLS Autodiscovery Support on a Route Reflector

By default, routes received from an internal BGP (iBGP) peer are not sent to another iBGP peer unless a full mesh configuration is formed between all BGP devices within an autonomous system (AS). This results in scalability issues. Using Border Gateway Protocol (BGP) route reflectors leads to much higher levels of scalability. Configuring a route reflector allows a device to advertise or reflect the iBGP learned routes to other iBGP speakers.

Virtual Private LAN Service (VPLS) Autodiscovery supports BGP route reflectors. A BGP route reflector can be used to reflect BGP VPLS prefixes without VPLS being explicitly configured on the route reflector.

A route reflector does not participate in autodiscovery; that is, no pseudowires are set up between the route reflector and the PE devices. A route reflector reflects VPLS prefixes to other PE devices so that these PE devices do not need to have a full mesh of BGP sessions. The network administrator configures only the BGP VPLS address family on a route reflector. For an example configuration of VPLS Autodiscovery support on a route reflector, see the "Example: BGP VPLS Autodiscovery Support on Route Reflector" section.

N-PE Access to VPLS Using MST

When a Virtual Private LAN Service (VPLS) network uses multihoming (network-facing PE [N-PE] VPLS redundancy) to prevent a single point of failure of an N-PE device, a bridging loop is introduced. One of the N-PE devices can be set as a Multiple Spanning Tree (MST) root to break the loop. In most cases, the two N-PE devices are also separated by a distance that makes direct physical link impossible. You can configure a virtual link (usually through the same VPLS core network) between the two N-PE devices to pass an MST bridge protocol data unit (BPDU) for path calculation, break the loop, and maintain convergence. The virtual link is created using a special pseudowire between the active and redundant N-PE devices.

While setting up an MST topology for a VPLS PE device, ensure the following:

- The **spanning-tree mode mst** command is enabled on all PE devices (N-PE and user-facing PE [U-PE]) participating in the MST topology.
- A special pseudowire is configured between the two N-PE devices, and these two devices are in the up state.
- The special pseudowire is a manually created virtual forwarding instance (VFI).
- The configuration (inlcuding the MST instance, the Ethernet virtual circuit [EVC], and the VLAN) on all PE devices is the same.
- One of the N-PE devices, and not one of the U-PE devices, is the root for the MST instance.
- The name and revision for the MST configuration are configured to synchronize with the standby Route Processor (RP).

How to Configure VPLS Autodiscovery BGP Based

Enabling VPLS Autodiscovery BGP Based

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

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Device(config)# l2 vfi vpls1 autodiscovery`	Enables VPLS Autodiscovery on a PE device and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits L2 VFI configuration mode.

Enabling VPLS Autodiscovery BGP Based using the commands associated with the L2VPN Protocol-Based CLIs feature

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. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**ldp** | **bgp**}
6. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes an L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**ldp** | **bgp**} **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp`	Enables the VPLS Autodiscovery: BGP Based feature on the PE device.
Step 6	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits L2 VFI configuration mode.

Configuring VPLS BGP Signaling

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**bgp** | **ldp**} [**template** *template-name*]
6. **ve id** *ve-id*

7. **ve range** *ve-range*
8. **exit**
9. **exit**
10. **router bgp** *autonomous-system-number*
11. **bgp graceful-restart**
12. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
13. **address-family l2vpn** [**vpls**]
14. **neighbor** *ip-address* **activate**
15. **neighbor** *ip-address* **send-community** [**both** | **standard** | **extended**]
16. **neighbor** *ip-address* **suppress-signaling-protocol ldp**
17. **end**
18. **show bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *name* **Example:** `Device(config)# l2vpn vfi context vfi1`	Establishes a L2VPN virtual forwarding interface (VFI) between two or more separate networks and enters Layer 2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 100`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**bgp** | **ldp**} [**template** *template-name*] **Example:** `Device(config-vfi)# autodiscovery bgp signaling bgp`	Enables BGP signaling and discovery or LDP signaling and enters L2VPN VFI autodiscovery configuration mode. **Note** For the VPLS BGP Signaling feature use the **autodiscovery bgp signaling bgp** command.

	Command or Action	Purpose
Step 6	**ve id** *ve-id* **Example:** `Device(config-vfi-autodiscovery)# ve id 1001`	Specifies the VPLS endpoint (VE) device ID value. The VE ID identifies a VFI within a VPLS service. The VE device ID value is from 1 to 16384.
Step 7	**ve range** *ve-range* **Example:** `Device(config-vfi-autodiscovery)# ve range 12`	Specifies the VE device ID range value. The VE range overrides the minimum size of VE blocks. The default minimum size is 10. Any configured VE range must be higher than 10.
Step 8	**exit** **Example:** `Device(config-vfi-autodiscovery)# exit`	Exits L2VPN VFI autodiscovery configuration mode and enters L2VPN VFI configuration mode.
Step 9	**exit** **Example:** `Device(config-vfi)# exit`	Exits L2VPN VFI configuration mode and enters global configuration mode.
Step 10	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode to create or configure a BGP routing process.
Step 11	**bgp graceful-restart** **Example:** `Device(config-router)# bgp graceful-restart`	Enables the BGP graceful restart capability and BGP nonstop forwarding (NSF) awareness.
Step 12	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 remote-as 100`	Configures peering with a BGP neighbor in the specified autonomous system.
Step 13	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that VPLS endpoint provisioning information is to be distributed to BGP peers. In this example, an L2VPN VPLS address family session is created.

	Command or Action	Purpose
Step 14	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the neighbor to exchange information for the L2VPN VPLS address family with the local device.
Step 15	**neighbor** *ip-address* **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 16	**neighbor** *ip-address* **suppress-signaling-protocol ldp** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 suppress-signaling-protocol ldp`	Suppresses LDP signaling and enables BGP signaling. • In this example LDP signaling is suppressed (and BGP signaling enabled) for the neighbor at 10.10.10.1.
Step 17	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 18	**show bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:** `Device# show bgp l2vpn vpls all`	(Optional) Displays information about the L2VPN VPLS address family.

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** {*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**
15. **show vfi**
16. **show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 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.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:**	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.

	Command or Action	Purpose
	`Device(config-router)# neighbor 10.10.10.1 remote-as 65000`	• 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. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 update-source loopback1`	(Optional) Configures a device to select a specific source or interface to receive routing table updates. • This example uses a loopback interface. The advantage of this configuration is that the loopback interface is not affected by the effects of a flapping interface.
Step 8	Repeat Steps 6 and 7 to configure other BGP neighbors.	—
Step 9	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers. • In this example, an L2VPN VPLS address family session is created.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 12	Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.	—
Step 13	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.

	Command or Action	Purpose
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.
Step 15	**show vfi** **Example:** `Device# show vfi`	Displays information about the configured VFI instances.
Step 16	**show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:** `Device# show ip bgp l2vpn vpls all`	Displays information about the L2VPN VPLS address family.

Customizing the VPLS Autodiscovery Settings

Several commands allow you to customize the Virtual Private LAN Service (VPLS) environment. You can specify identifiers for the VPLS domain, the route distinguisher (RD), the route target (RT), and the provider edge (PE) device. Perform this task to customize these identifiers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **autodiscovery**
4. **vpn id** *vpn-id*
5. **vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
6. **rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
7. **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
8. **auto-route-target**
9. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Device(config)# l2 vfi vpls1 autodiscovery`	Enables VPLS Autodiscovery on the PE device and enters Layer 2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# vpls-id 5:300`	(Optional) Assigns an identifier to the VPLS domain. • This command is optional because VPLS Autodiscovery automatically generates a VPLS ID using the Border Gateway Protocol (BGP) autonomous system (AS) number and the configured VFI VPN ID. You can use this command to change the automatically generated VPLS ID. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 6	**rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# rd 2:3`	(Optional) Specifies the RD to distribute endpoint information. • This command is optional because VPLS Autodiscovery automatically generates an RD using the BGP autonomous system number and the configured VFI VPN ID. You can use this command to change the automatically generated RD. • There are two formats for configuring the route distinguisher argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 7	**route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# route-target 600:2222`	(Optional) Specifies the RT. • This command is optional because VPLS Autodiscovery automatically generates an RT using the lower 6 bytes of the RD and the VPLS ID. You can use this command to change the automatically generated RT. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*)

	Command or Action	Purpose
		format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 8	**auto-route-target** **Example:** `Device(config-vfi)# auto-route-target`	(Optional) Enables the automatic generation of a RT.
Step 9	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits Layer 2 VFI configuration mode.

Configuring BGP to Enable VPLS Autodiscovery using the commands associated with the L2VPN Protocol-Based CLIs feature

The BGP 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** {*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**
15. **show l2vpn vfi**
16. **show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 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.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 remote-as 65000`	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.

	Command or Action	Purpose
		• In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.10.10.1 update-source loopback1`	(Optional) Configures a device to select a specific source or interface to receive routing table updates. • This example uses a loopback interface. The advantage of this configuration is that the loopback interface is not affected by the effects of a flapping interface.
Step 8	Repeat Steps 6 and 7 to configure other BGP neighbors.	—
Step 9	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the L2VPN address family and enters address family configuration mode. • The optional **vpls** keyword specifies that the VPLS endpoint provisioning information is to be distributed to BGP peers. • In this example, an L2VPN VPLS address family session is created.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.10.10.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.
Step 12	Repeat Steps 10 and 11 to activate other BGP neighbors under an L2VPN address family.	—
Step 13	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 15	**show l2vpn vfi** **Example:** `Device# show l2vpn vfi`	Displays information about the Layer 2 VPN (L2VPN) virtual forwarding instances (VFI).
Step 16	**show ip bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*} **Example:** `Device# show ip bgp l2vpn vpls all`	Displays information about the L2VPN VPLS address family.

Customizing the VPLS Autodiscovery Settings using the commands associated with the L2VPN Protocol-Based CLIs feature

Several commands allow you to customize the Virtual Private LAN Service (VPLS) environment. You can specify identifiers for the VPLS domain, the route distinguisher (RD), the route target (RT), and the provider edge (PE) device. Perform this task to customize these identifiers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **autodiscovery bgp signaling** {**ldp** | **bgp**}
6. **vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
7. **rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
8. **route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}
9. **auto-route-target**
10. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls1`	Establishes a L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 10`	Configures a VPN ID for the VPLS domain.
Step 5	**autodiscovery bgp signaling** {**ldp** | **bgp**} **Example:** `Device(config-vfi)# autodiscovery bgp signaling ldp`	Enables the VPLS Autodiscovery: BGP Based feature on the PE device.
Step 6	**vpls-id** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# vpls-id 5:300`	(Optional) Assigns an identifier to the VPLS domain. • This command is optional because VPLS Autodiscovery automatically generates a VPLS ID using the Border Gateway Protocol (BGP) autonomous system (AS) number and the configured VFI VPN ID. You can use this command to change the automatically generated VPLS ID. • There are two formats for configuring the VPLS ID argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 7	**rd** {*autonomous-system-number*:*nn* | *ip-address*:*nn*} **Example:** `Device(config-vfi)# rd 2:3`	(Optional) Specifies the RD to distribute endpoint information. • This command is optional because VPLS Autodiscovery automatically generates an RD using the BGP autonomous system number and the configured VFI VPN ID. You can use this command to change the automatically generated RD. • There are two formats for configuring the route distinguisher argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 8	**route-target** [**import** | **export** | **both**] {*autonomous-system-number*:*nn* | *ip-address*:*nn*}	(Optional) Specifies the RT.

	Command or Action	Purpose
	Example: `Device(config-vfi)# route-target 600:2222`	• This command is optional because VPLS Autodiscovery automatically generates an RT using the lower 6 bytes of the RD and the VPLS ID. You can use this command to change the automatically generated RT. • There are two formats for configuring the route target argument. It can be configured in the *autonomous-system-number:network number* (*ASN:nn*) format, as shown in the example, or it can be configured in the *IP-address:network number* format (*IP-address:nn*).
Step 9	**auto-route-target** **Example:** `Device(config-vfi)# auto-route-target`	(Optional) Enables the automatic generation of a RT.
Step 10	**end** **Example:** `Device(config-vfi)# end`	Exits L2 VFI configuration mode and returns to privileged EXEC mode. • Commands take effect after the device exits Layer 2 VFI configuration mode.

Configuring MST on VPLS N-PE Devices

A network-facing PE (N-PE) device is the root bridge for a Multiple Spanning Tree (MST) instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi-name* **manual**
4. **vpn id** *vpn-id*
5. **forward permit l2protocol all**
6. **neighbor** *peer-N-PE-ip-address* **encapsulation mpls**
7. **exit**
8. **spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**]
9. **spanning-tree mst configuration**
10. **name** *name*
11. **revision** *version*
12. **instance** *instance-id* **vlan** *vlan-range*
13. **end**
14. **show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2 vfi** *vfi-name* **manual** **Example:** `Device(config)# l2 vfi vpls-mst manual`	Creates a Layer 2 virtual forwarding instance (VFI) and enters Layer 2 VFI manual configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 4000`	Sets or updates the VPN ID on a VPN routing and forwarding (VRF) instance.
Step 5	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Defines the VPLS pseudowire that is used to transport the bridge protocol data unit (BPDU) information between two N-PE devices.
Step 6	**neighbor** *peer-N-PE-ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls`	Specifies the type of tunnel signaling and encapsulation mechanism for each VPLS peer.
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Exits Layer 2 VFI manual configuration mode and returns to global configuration mode.
Step 8	**spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**] **Example:** `Device(config)# spanning-tree mode mst`	Switches between MST, Per-VLAN Spanning Tree+ (PVST+), and Rapid-PVST+ modes.
Step 9	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration mode.
Step 10	**name** *name* **Example:** `Device(config-mst)# name cisco`	Sets the name for the MST region.

	Command or Action	Purpose
Step 11	**revision** *version* **Example:** `Device(config-mst)# revision 11`	Sets the revision number for the MST configuration.
Step 12	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 100`	Maps a VLAN or a group of VLANs to an MST instance.
Step 13	**end** **Example:** `Device(config-mst)# end`	Exits MST configuration mode and enters privileged EXEC mode.
Step 14	**show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]] **Example:** `Device# show spanning-tree mst 1`	Displays information about the MST configuration.

Configuring MST on VPLS N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

A network-facing PE (N-PE) device is the root bridge for a Multiple Spanning Tree (MST) instance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2vpn vfi context** *vfi-name*
4. **vpn id** *vpn-id*
5. **forward permit l2protocol all**
6. **neighbor** *peer-N-PE-ip-address* **encapsulation mpls**
7. **exit**
8. **spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**]
9. **spanning-tree mst configuration**
10. **name** *name*
11. **revision** *version*
12. **instance** *instance-id* **vlan** *vlan-range*
13. **end**
14. **show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**l2vpn vfi context** *vfi-name* **Example:** `Device(config)# l2vpn vfi context vpls-mst`	Establishes an L2VPN VFI context and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 4000`	Sets or updates the VPN ID on a VPN routing and forwarding (VRF) instance.
Step 5	**forward permit l2protocol all** **Example:** `Device(config-vfi)# forward permit l2protocol all`	Defines the VPLS pseudowire that is used to transport the bridge protocol data unit (BPDU) information between two N-PE devices.
Step 6	**neighbor** *peer-N-PE-ip-address* **encapsulation mpls** **Example:** `Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls`	Specifies the type of tunnel signaling and encapsulation mechanism for each VPLS peer.
Step 7	**exit** **Example:** `Device(config-vfi)# exit`	Exits Layer 2 VFI manual configuration mode and returns to global configuration mode.
Step 8	**spanning-tree mode** [**mst** | **pvst** | **rapid-pvst**] **Example:** `Device(config)# spanning-tree mode mst`	Switches between MST, Per-VLAN Spanning Tree+ (PVST+), and Rapid-PVST+ modes.
Step 9	**spanning-tree mst configuration** **Example:**	Enters MST configuration mode.

	Command or Action	Purpose
	`Device(config)# spanning-tree mst configuration`	
Step 10	**name** *name* **Example:** `Device(config-mst)# name cisco`	Sets the name for the MST region.
Step 11	**revision** *version* **Example:** `Device(config-mst)# revision 11`	Sets the revision number for the MST configuration.
Step 12	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 100`	Maps a VLAN or a group of VLANs to an MST instance.
Step 13	**end** **Example:** `Device(config-mst)# end`	Exits MST configuration mode and enters privileged EXEC mode.
Step 14	**show spanning-tree mst** [*instance-id* [**detail**] [*interface*] | **configuration** [**digest**] | **detail** | **interface** *type number* [**detail**]] **Example:** `Device# show spanning-tree mst 1`	Displays information about the MST configuration.

Configuration Examples for VPLS Autodiscovery BGP Based

The following examples show the configuration of a network that uses VPLS Autodiscovery:

Example: Enabling VPLS Autodiscovery BGP Based

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls1 autodiscovery
Device(config-vfi)# vpn id 10
Device(config-vfi)# exit
```

Example: Enabling VPLS Autodiscovery BGP Based Using Commands Associated with L2VPN Protocol-Based Feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls1
Device(config-vfi)# vpn id 10
Device(config-vfi)# autodiscovery bgp signaling ldp
Device(config-vfi)# exit
```

Example: Configuring BGP to Enable VPLS Autodiscovery

PE1

```
l2 router-id 10.1.1.1
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.1 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.1 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE2

```
l2 router-id 10.1.1.2
l2 vfi auto autodiscovery
 vpn id 100
!
 pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.2 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.2 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE3

```
l2 router-id 10.1.1.3
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.3 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.3 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
```

```
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 exit-address-family
```

Example: Configuring BGP to Enable VPLS Autodiscovery Using Commands Associated with L2VPN Protocol-Based Feature

PE1

```
l2vpn
 router-id 10.1.1.1
l2vpn vfi context auto
 vpn id 100
 autodiscovery bgp signaling ldp
!
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.1 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.1 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
```

```
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE2

```
l2vpn
 router-id 10.1.1.2
l2vpn vfi context auto
 vpn id 100
 autodiscovery bgp signaling ldp

!
 interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.2 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.2 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.3 remote-as 1
 neighbor 10.1.1.3 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.3 activate
 neighbor 10.1.1.3 send-community extended
 exit-address-family
```

PE3

```
l2vpn
 router-id 10.1.1.3
l2vpn vfi context auto
 vpn id 100
```

```
 autodiscovery bgp signaling ldp

!
interface pseudowire 1
 encapsulation mpls
 neighbor 33.33.33.33 1
!
interface Loopback1
 ip address 10.1.1.3 255.255.255.255
!
interface GigabitEthernet 0/0/1
 description Backbone interface
 ip address 192.168.0.3 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.0.255 area 0
!
router bgp 1
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 update-source Loopback1
 neighbor 10.1.1.2 remote-as 1
 neighbor 10.1.1.2 update-source Loopback1
!
 address-family ipv4
 no synchronization
 no auto-summary
 exit-address-family
 !
 address-family l2vpn vpls
 neighbor 10.1.1.1 activate
 neighbor 10.1.1.1 send-community extended
 neighbor 10.1.1.2 activate
 neighbor 10.1.1.2 send-community extended
 exit-address-family
```

Example: Customizing VPLS Autodiscovery Settings

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls1 autodiscovery
Device(config-vfi)# vpn id 10
Device(config-vfi)# vpls-id 5:300
Device(config-vfi)# rd 2:3
Device(config-vfi)# route-target 600:2222
Device(config-vfi)# end
```

Example: Customizing VPLS Autodiscovery Settings using the commands associated with the L2VPN Protocol-Based CLIs feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls1
```

```
Device(config-vfi)# vpn id 10
Device(config-vfi)# autodiscovery bgp signaling ldp
Device(config-vfi)# vpls-id 5:300
Device(config-vfi)# rd 2:3
Device(config-vfi)# route-target 600:2222
Device(config-vfi)# end
```

Example: Configuring MST on VPLS N-PE Devices

```
Device> enable
Device# configure terminal
Device(config)# l2 vfi vpls-mst manual
Device(config-vfi)# vpn id 4000
Device(config-vfi)# forward permit l2protocol all
Device(config-vfi)# neighbor 10.76.100.12 encapsulation mpls
Device(config-vfi)# exit
Device(config)# spanning-tree mode mst
Device(config)# spanning-tree mst configuration
Device(config-mst)# name cisco
Device(config-mst)# revision 11
Device(config-mst)# instance 1 vlan 100
Device(config-mst)# end
```

The following is sample output from the **show spanning-tree mst** command:

```
Device# show spanning-tree mst 1

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1                              // Root for MST instance
 1 with VLAN 100
Interface                     Role Sts Cost      Prio.Nbr Type
-----------                ---- -- ------ ---- --------------------------------
Gi1/0/0                       Desg FWD 20000     128.18   P2p   // Access interface
VPLS-MST                      Desg FWD 1         128.28   Shr   // Forward VFI
```

The following is sample output from the **show spanning-tree mst detail** command:

```
Device# show spanning-tree mst 1 detail

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1            // Root for MST instance 1 with VLAN 100
GigabitEthernet1/0/0 of MST1 is designated forwarding
Port info             port id         128.18  priority    128  cost       20000
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.18
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 40, received 5
VPLS-4000 of MST1 is designated forwarding
Port info             port id         128.28  priority    128  cost           1
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.28
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 28, received 26         // BPDU message exchange between N-PE devices
```

Example: Configuring MST on VPLS N-PE Devices using the commands associated with the L2VPN Protocol-Based CLIs feature

```
Device> enable
Device# configure terminal
Device(config)# l2vpn vfi context vpls-mst
Device(config-vfi)# vpn id 4000
Device(config-vfi)# forward permit l2protocol all
Device(config-vfi)# member 10.76.100.12 encapsulation mpls
Device(config-vfi)# exit
Device(config)# spanning-tree mode mst
Device(config)# spanning-tree mst configuration
Device(config-mst)# name cisco
Device(config-mst)# revision 11
Device(config-mst)# instance 1 vlan 100
Device(config-mst)# end
```

The following is sample output from the **show spanning-tree mst** command:

```
Device# show spanning-tree mst 1

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1                              // Root for MST instance
 1 with VLAN 100
Interface                       Role Sts Cost      Prio.Nbr Type
-----------                  --- -- ------ ---- -------------------------------
Gi1/0/0                         Desg FWD 20000     128.18   P2p   // Access interface
VPLS-MST                        Desg FWD 1         128.28   Shr   // Forward VFI
```

The following is sample output from the **show spanning-tree mst detail** command:

```
Device# show spanning-tree mst 1 detail

##### MST1    vlans mapped:   100
Bridge        address 0023.3380.f8bb  priority      4097  (4096 sysid 1)
Root          this switch for MST1            // Root for MST instance 1 with VLAN 100
GigabitEthernet1/0/0 of MST1 is designated forwarding
Port info             port id         128.18  priority    128  cost       20000
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.18
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 40, received 5
VPLS-4000 of MST1 is designated forwarding
Port info             port id         128.28  priority    128  cost           1
Designated root       address 0023.3380.f8bb  priority   4097  cost           0
Designated bridge     address 0023.3380.f8bb  priority   4097  port id   128.28
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 28, received 26         // BPDU message exchange between N-PE devices
```

Example: BGP VPLS Autodiscovery Support on Route Reflector

In the following example, a host named PE-RR (indicating Provider Edge-Route Reflector) is configured as a route reflector that is capable of reflecting Virtual Private LAN Service (VPLS) prefixes. The VPLS address family is configured using the **address-family l2vpn vpls** command.

```
hostname PE-RR
!
router bgp 1
 bgp router-id 10.1.1.3
```

```
 no bgp default route-target filter
 bgp log-neighbor-changes
neighbor iBGP-PEERS peer-group
neighbor iBGP-PEERS remote-as 1
neighbor iBGP-PEERS update-source Loopback1
neighbor 10.1.1.1 peer-group iBGP-PEERS
neighbor 10.1.1.2 peer-group iBGP-PEERS
!
address-family l2vpn vpls
  neighbor iBGP-PEERS send-community extended
  neighbor iBGP-PEERS route-reflector-client
  neighbor 10.1.1.1 peer-group iBGP-PEERS
  neighbor 10.1.1.2 peer-group iBGP-PEERS
exit-address-family
```

Additional References for VPLS Autodiscovery BGP Based

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
MPLS commands	Multiprotocol Label Switching Command Reference

Standards and RFCs

Standard/RFC	Title
draft-ietf-l2vpn-signaling-08.txt	*Provisioning, Autodiscovery, and Signaling in L2VPNs*
draft-ietf-l2vpn-vpls-bgp-08.8	*Virtual Private LAN Service (VPLS) Using BGP for Autodiscovery and Signaling*
draft-ietf-mpls-lsp-ping-03.txt	*Detecting MPLS Data Plane Failures*
draft-ietf-pwe3-vccv-01.txt	*Pseudo-Wire (PW) Virtual Circuit Connection Verification (VCCV)*
RFC 3916	*Requirements for Pseudo-wire Emulation Edge-to-Edge (PWE3)*
RFC 3981	*Pseudo Wire Emulation Edge-to-Edge Architecture*
RFC 6074	Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)
RFC 4761	Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling

MIBs

MIB	MIBs Link
• CISCO-IETF-PW-ATM-MIB (PW-ATM-MIB) • CISCO-IETF-PW-ENET-MIB (PW-ENET-MIB) • CISCO-IETF-PW-FR-MIB (PW-FR-MIB) • CISCO-IETF-PW-MIB (PW-MIB) • CISCO-IETF-PW-MPLS-MIB (PW-MPLS-MIB)	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/techsupport

Feature Information for VPLS Autodiscovery BGP Based

Table 3: Feature Information for VPLS Autodiscovery BGP Based

Feature Name	Releases	Feature Information
VPLS Autodiscovery BGP Based	Cisco IOS XE Release 3.7S Cisco IOS Release 15.1(1)SY	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.