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

Configuring VPLS

The following sections provide information about how to configure VPLS.

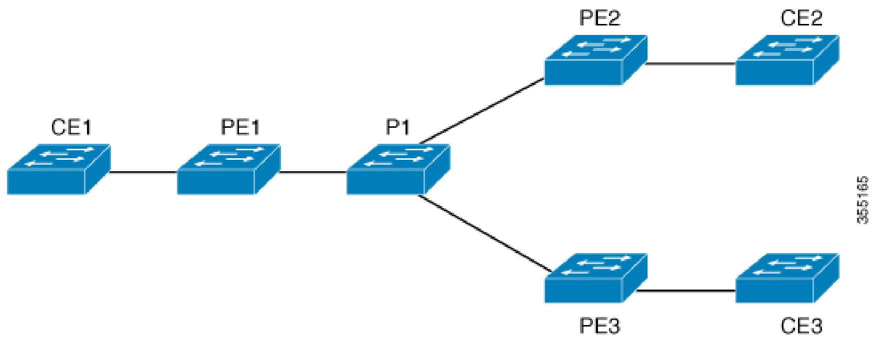
Information About VPLS

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 Service (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.

Figure 1: VPLS Topology



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.

Restrictions for VPLS

- Layer 2 protocol tunneling configuration is not supported
- Integrated Routing and Bridging (IRB) configuration is not supported.
- Virtual Circuit Connectivity Verification (VCCV) ping with explicit null is not supported.
- The switch is supported if configured only as a spoke in hierarchical Virtual Private LAN Services (VPLS) and not as a hub.
- Layer 2 VPN interworking functions are not supported.
- **ip unnumbered** command is not supported in Multiprotocol Label Switching (MPLS) configuration.
- Virtual Circuit (VC) statistics are not displayed for flood traffic in the output of **show mpls l2 vc vcid detail** command.
- Dot1q tunnel configuration is not supported in the attachment circuit.

Configuring Layer 2 PE Device Interfaces to CE Devices

You must configure Layer 2 PE device interfaces to CE devices. You can either configure 802.1Q trunks on the PE device for tagged traffic from a CE device or configure 802.1Q access ports on the PE device for untagged traffic from a CE device. The following sections provides configuration information for both.

Configuring 802.1Q Trunks on a PE Device for Tagged Traffic from a CE Device

To configure 802.1Q trunks on a PE device, perform this procedure:

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	**interface** *interface-id* **Example:** `Device(config)# interface TenGigabitEthernet1/0/24`	Defines the interface to be configured as a trunk, and enters interface configuration mode.
Step 4	**no ip address** *ip_address mask* [**secondary**] **Example:**	Disables IP processing and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config-if)# no ip address`	
Step 5	**switchport** **Example:** `Device(config-if)# switchport`	Modifies the switching characteristics of the Layer 2 switched interface.
Step 6	**switchport trunk encapsulation dot1q** **Example:** `Device(config-if)# switchport trunk encapsulation dot1q`	Sets the switch port encapsulation format to 802.1Q.
Step 7	**switchport trunk allow vlan** *vlan_ID* **Example:** `Device(config-if)# switchport trunk allow vlan 2129`	Sets the list of allowed VLANs.
Step 8	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface to a trunking VLAN Layer 2 interface.
Step 9	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring 802.1Q Access Ports on a PE Device for Untagged Traffic from a CE Device

To configure 802.1Q access ports on a PE device, perform this procedure:

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	**interface** *interface-id* **Example:** `Device(config)# interface TenGigabitEthernet1/0/24`	Defines the interface to be configured as a trunk, and enters interface configuration mode.
Step 4	**no ip address** *ip_address mask* [**secondary**] **Example:** `Device(config-if)# no ip address`	Disables IP processing.
Step 5	**switchport** **Example:** `Device(config-if)# switchport`	Modifies the switching characteristics of the Layer 2 switched interface.
Step 6	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the interface type to nontrunking and nontagged single VLAN Layer 2 interface.
Step 7	**switchport access vlan** *vlan_ID* **Example:** `Device(config-if)# switchport access vlan 2129`	Sets the VLAN when the interface is in access mode.
Step 8	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Layer 2 VLAN Instances on a PE Device

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

To configure Layer 2 VLAN instance on a PE device, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 2129`	Configures a specific VLAN.
Step 4	**interface vlan** *vlan-id* **Example:** `Device(config-vlan)# interface vlan 2129`	Configures an interface on the VLAN.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Returns to privileged EXEC mode.

Configuring MPLS on a PE Device

To configure MPLS on a PE device, perform this procedure:

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	**mpls ip** **Example:** `Device(config)# mpls ip`	Configures MPLS hop-by-hop forwarding.
Step 4	**mpls label protocol ldp** **Example:**	Specifies the default Label Distribution Protocol (LDP) for a platform.

	Command or Action	Purpose
	`Device(config)# mpls label protocol ldp`	
Step 5	**mpls ldp logging neighbor-changes** **Example:** `Device(config)# mpls ldp logging neighbor-changes`	(Optional) Determines logging neighbor changes.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring VFI on a PE Device

The 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 device.

To configure VFI and associated VCs on the PE device, perform this procedure:

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 2129 manual`	Enables the Layer 2 VFI manual configuration mode.
Step 4	**vpn id** *vpn-id* **Example:** `Device(config-vfi)# vpn id 2129`	Configures a VPN ID for a VPLS domain. The emulated VCs bound to this Layer 2 virtual routing and forwarding (VRF) use this VPN ID for signaling. **Note** *vpn-id* is the same as *vlan-id*.

	Command or Action	Purpose
Step 5	**neighbor** *router-id* {**encapsulation mpls**} **Example:** `Device(config-vfi)# neighbor remote-router-id encapsulation mpls`	Specifies the remote peering router ID and the tunnel encapsulation type or the pseudowire (PW) property to be used to set up the emulated VC.
Step 6	**end** **Example:** `Device(config-vfi)# end`	Returns to privileged EXEC mode.

Associating the Attachment Circuit with the VFI on the PE Device

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

To associate the attachment circuit with the VFI, perform this procedure:

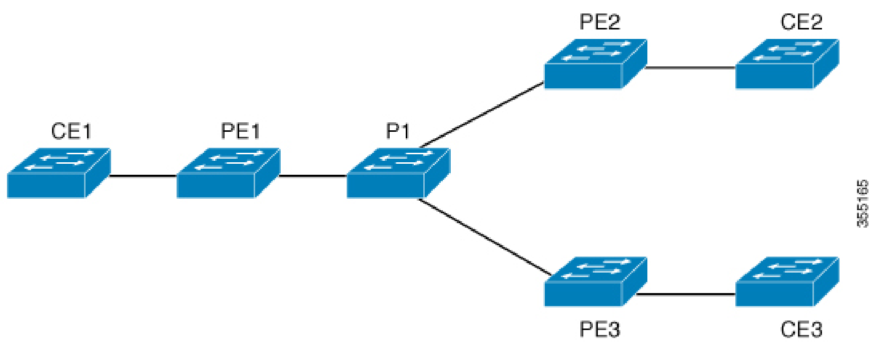
Procedure

	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	**interface vlan** *vlan-id* **Example:** `Device(config)# interface vlan 2129`	Creates or accesses a dynamic switched virtual interface (SVI). **Note** *vlan-id* is the same as *vpn-id*.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing. (You can configure a Layer 3 interface for the VLAN if you need to configure an IP address.)
Step 5	**xconnect vfi** *vfi-name* **Example:** `Device(config-if)# xconnect vfi 2129`	Specifies the Layer 2 VFI that you are binding to the VLAN port.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuration Examples for VPLS

Figure 2: VPLS Topology



<table>
<tr><th>PE1 Configuration</th><th>PE2 Configuration</th></tr>
<tr><td><pre>
pseudowire-class vpls2129
 encapsulation mpls
!
l2 vfi 2129 manual
 vpn id 2129
 neighbor 44.254.44.44 pw-class vpls2129

 neighbor 188.98.89.98 pw-class vpls2129
!
interface TenGigabitEthernet1/0/24
 switchport trunk allowed vlan 2129
 switchport mode trunk
!
interface Vlan2129
 no ip address
 xconnect vfi 2129
!
</pre></td><td><pre>
pseudowire-class vpls2129
 encapsulation mpls
 no control-word
!
l2 vfi 2129 manual
 vpn id 2129
 neighbor 1.1.1.72 pw-class vpls2129
 neighbor 188.98.89.98 pw-class vpls2129
!
interface TenGigabitEthernet1/0/47
 switchport trunk allowed vlan 2129
 switchport mode trunk
end
!
interface Vlan2129
 no ip address
 xconnect vfi 2129
!
</pre></td></tr>
</table>

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 sent: 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
Status sent to network peer             : No fault
        Status received from network peer       : No fault
        Adjacency status of remote peer         : No fault
  Sequencing: receive disabled, send disabled
  Bindings
    Parameter    Local                          Remote
    ------------ ------------------------------
------------------------------
    Label        512                            17
    Group ID     n/a                            0
    Interface

    MTU          1500                           1500
    Control word off                            off
    PW type      Ethernet                       Ethernet
    VCCV CV type 0x02                           0x02
                   LSPV [2]                       LSPV [2]

    VCCV CC type 0x06                           0x06
                   RA [2], TTL [3]               RA [2], TTL [3]
    Status TLV   enabled                        supported
  SSO Descriptor: 44.254.44.44/2129, local label: 512
  Dataplane:
    SSM segment/switch IDs: 20498/20492 (used), PWID: 2
  Rx Counters
    0 input transit packets, 0 bytes
    0 drops, 0 seq err
  Tx Counters
    0 output transit packets, 0 bytes
    0 drops
```

Configuring VPLS BGP-based Autodiscovery

The following sections provide information about how to configure 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.

Enabling VPLS BGP-based Autodiscovery

To enabling VPLS BGP-based autodiscovery, perform this procedure:

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 2128 autodiscovery`	Enables VPLS autodiscovery on a PE device and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn-id* **Example:**	Configures a VPN ID for the VPLS domain.

	Command or Action	Purpose
	`Device(config-vfi)# vpn id 2128`	
Step 5	**end** **Example:** `Device(config-vfi)# end`	Returns to privileged EXEC mode.

Configuring BGP to Enable VPLS Autodiscovery

To configure BGP to enable VPLS autodiscovery, perform this procedure:

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 1000`	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** command unless you configure the **no bgp default ipv4-unicast** command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.

	Command or Action	Purpose
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 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.
Step 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.
Step 8	Repeat Steps 6 and 7 to configure other BGP neighbors.	Exits interface configuration mode.
Step 9	**address-family l2vpn** [**vpls**] **Example:** `Device(config-router)# address-family l2vpn vpls`	Specifies the Layer 2 VPN 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.
Step 10	**neighbor** { *ip-address* | *peer-group-name* } **activate** **Example:** `Device(config-router-af)# neighbor 44.254.44.44 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 44.254.44.44 send-community both`	Specifies that a communities attribute should be sent to a BGP neighbor.

	Command or Action	Purpose
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.

Configuration Examples for VPLS BGP-AD

PE Configuration

```
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 :

```
VLAN  MAC                 Type      Seq#   macHandle          siHandle
      diHandle              *a_time *e_time   ports
2000  2852.6134.05c8    0X8002    0      0xffbba312c8       0xffbb9ef938
      0x5154              0         0        Vlan2000
2000  0000.0078.9012    0X1       32627  0xffbb665ec8       0xffbb60b198
      0xffbb653f98        300       278448   Port-channel11
2000  2852.6134.0000    0X1       32651  0xffba15e1a8       0xff454c2328
      0xffbb653f98        300       63       Port-channel11
```

```
2000  0000.0012.3456   0X2000001 32655  0xffba15c508       0xff44f9ec98
      0x0                300      1        2000:33.33.33.33
Total Mac number of addresses:: 4
*a_time=aging_time(secs)  *e_time=total_elapsed_time(secs)
Type:
MAT_DYNAMIC_ADDR     0x1       MAT_STATIC_ADDR       0x2
MAT_CPU_ADDR         0x4       MAT_DISCARD_ADDR      0x8
MAT_ALL_VLANS        0x10      MAT_NO_FORWARD        0x20
MAT_IPMULT_ADDR      0x40      MAT_RESYNC            0x80
MAT_DO_NOT_AGE       0x100     MAT_SECURE_ADDR       0x200
MAT_NO_PORT          0x400     MAT_DROP_ADDR         0x800
MAT_DUP_ADDR         0x1000    MAT_NULL_DESTINATION  0x2000
MAT_DOT1X_ADDR       0x4000    MAT_ROUTER_ADDR       0x8000
MAT_WIRELESS_ADDR    0x10000   MAT_SECURE_CFG_ADDR   0x20000
MAT_OPQ_DATA_PRESENT 0x40000   MAT_WIRED_TUNNEL_ADDR 0x80000
MAT_DLR_ADDR         0x100000  MAT_MRP_ADDR          0x200000
MAT_MSRP_ADDR        0x400000  MAT_LISP_LOCAL_ADDR   0x800000
MAT_LISP_REMOTE_ADDR 0x1000000 MAT_VPLS_ADDR         0x2000000
```

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
Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1000:2128
*>   1000:2128:1.1.1.72/96
                0.0.0.0                           32768 ?
*>i  1000:2128:44.254.44.44/96
                44.254.44.44             0    100      0 ?
```

Feature History for VPLS and VPLS BGP-Based Autodiscovery

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Everest 16.5.1a	Configuring VPLS and VPLS BGP-based Autodiscovery	VPLS enables enterprises to link together their Ethernet-based LANs from multiple sites via the infrastructure provided by their service provider. VPLS Autodiscovery enables each PE device to discover other PE devices that are part of the same VPLS domain.

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/

http://www.cisco.com/go/cfn.