Configuring A-VPLS

This chapter describes how to configure Layer 2 Virtual Private Networks (L2VPN) Advanced Virtual Private LAN Services (A-VPLS). Release 12.2(33)SXI4 and later releases support A-VPLS.

Note For complete syntax and usage information for the commands used in this chapter, see the Cisco IOS Master Command List, at this URL:

http://www.cisco.com/en/US/docs/ios/mcl/allreleasemcl/all_book.html

This chapter consists of these sections:

Tip For additional information about Cisco Catalyst 6500 Series Switches (including configuration examples and troubleshooting information), see the documents listed on this page:

http://www.cisco.com/en/US/products/hw/switches/ps708/tsd_products_support_series_home.html

Participate in the Technical Documentation Ideas forum

Understanding A-VPLS

A-VPLS introduces the following enhancements to VPLS:

- Ability to load-balance traffic at the provider edge (PE) among multiple equal-cost core-facing paths and at core interfaces using flow labels.
- Support for redundant PE routers.

A-VPLS uses the Flow Aware Transport (FAT) Pseudowire feature to achieve PE redundancy and load-balancing on both PE and core routers. FAT pseudowires are used to load-balance traffic in the core when equal cost multipaths are used. The PE router adds an additional MPLS Label to the each packet (the flow label). Each flow has a unique flow label. For more information about FAT pseudowires, see PWE3 Internet-Draft *Flow Aware Transport of MPLS Pseudowires* (draft-bryant-filsfils-fat-pw).

Restrictions for A-VPLS

- Release 12.2(33)SXJ1 and later releases support configuration of ES+ module ports as MPLS PE core-facing ports that carry A-VPLS traffic.
- Release 12.2(33)SXI4a and later releases support A-VPLS on these core facing port types in a 7600-SIP-400:
  - Gigabit and 10-Gigabit Ethernet SPAs (2X1GE-V1, 2X1GE-V2 and 1X10GE-V2 SPA)
  - Packet over Sonet (POS) SPAs (2XOC3, 4XOC3, 1XOC12 and 1XOC48)
- Release 12.2(33)SXI4a and later releases support these types of configurations:
  - MPLS core with configuration of PE routers through the **neighbor** command under transport vpls mode.
  - MPLS core with configuration of PE routers through MPLS traffic engineering tunnels using explicit paths.
  - IP core with configuration of PE routers through MPLS over GRE tunnels.

  Other configuration methods, including use of the **route-via** command, BGP autodiscovery, or explicit VLAN assignment to a PE egress port, are not supported.
- A-VPLS supports the following:
  - In switches without an ES+ line card:
    - —Up to 32 EtherChannel port-channel interfaces.
      (ES+ line cards do not support port-channel interfaces)
    - —Up to 60 VPLS neighbors, minus the number of neighbors configured with the **load-balance flow** command.
  - In switches with an ES+ line card (with or without a 7600-SIP-400):
    - —Up to 30 EtherChannel port-channel interfaces.
      (ES+ line cards do not support port-channel nterfaces)
    - —Up to 30 VPLS neighbors, minus the number of neighbors configured with the **load-balance flow** command.
- A-VPLS requires nonstop forwarding and stateful switchover.
- A-VPLS works with following:
  - MPLS Traffic Engineering tunnels that are configured with explicit paths.
  - Generic Routing Encapsulation (GRE tunnels) that are configured with static routes to the tunnel destination.

  For information about MPLS traffic engineering and GRE tunnels, see the following documents:
  - *MPLS Traffic Engineering and Enhancements*
  - *Implementing Tunnels*
- The **ping** and **traceroute** commands that support the Any Transport over MPLS Virtual Circuit Connection Verification (VCCV) feature are not supported over FAT pseudowires.
- The VPLS Autodiscovery feature is not supported with A-VPLS.
- Load-balancing is not supported in the core routers when the core uses IP to transport packets.

Configuring A-VPLS

The following sections explain how to configure A-VPLS:

- Enabling Load-Balancing with ECMP and FAT Pseudowires, page 33-3 (Required)
- Enabling Port-Channel Load-Balancing, page 33-4 (Required)
- Explicitly Specifying the PE Routers As Part of Virtual Ethernet Interface Configuration, page 33-4 (Optional)
- Configuring an MPLS Traffic Engineering Tunnel, page 33-5 (Optional)
- Configuring a GRE Tunnel, page 33-6 (Optional)

Enabling Load-Balancing with ECMP and FAT Pseudowires

The following steps explain how to configure load-balancing on the provider edge (PE) routers, which enables it on the core P routers. No configuration is required on the core P routers.

To enable load-balancing on the edge routers, issue the **load-balance flow** command. The load-balancing rules are configured through the **port-channel load-balance** command parameters (see the "Enabling Port-Channel Load-Balancing" section on page 33-4).

To enable core load-balancing, issue the **flow-label enable** command on both PE routers. You must issue the **load-balance flow** command with the **flow-label enable** command.

	Command	Purpose
Step 1	`Router> enable`	Enables privileged EXEC mode (enter your password if prompted).
Step 2	`Router# configure terminal`	Enters global configuration mode.
Step 3	`Router(config)# pseudowire-class` *name*	Establishes a pseudowire class with a name that you specify and enters pseudowire class configuration mode.
Step 4	`Router(config-pw)# encapsulation mpls`	Specifies the MPLS tunneling encapsulation type.
Step 5	`Router(config-pw)# load-balance flow`	Enables load-balancing on ECMPs.
Step 6	`Router(config-pw)# flow-label enable`	Enables the imposition and disposition of flow labels for the pseudowire.
Step 7	`Router(config-pw)# end`	Exits pseudowire class configuration mode and enters privileged EXEC mode.

Enabling Port-Channel Load-Balancing

The following task explains how to enable port channel load-balancing, which sets the load-distribution method among the ports in the bundle. If the **port-channel load-balance** command is not configured, load-balancing occurs with default parameters.

	Command	Purpose
Step 1	`Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	`Router# configure terminal`	Enters global configuration mode.
Step 3	`Router(config)# port-channel load-balance` *method*	Specifies the load distribution method among the ports in a bundle.
Step 4	`Router(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.

Explicitly Specifying the PE Routers As Part of Virtual Ethernet Interface Configuration

There are several ways to specify the route through which traffic should pass.

- Explicitly specify the PE routers as part of the virtual Ethernet interface configuration
- Configure an MPLS Traffic Engineering tunnel
- Configure a GRE tunnel

The following task explains how to explicitly specify the PE routers as part of the virtual Ethernet interface configuration.

	Command	Purpose
Step 1	`Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	`Router# configure terminal`	Enters global configuration mode.
Step 3	`Router(config)# interface virtual-ethernet` *num*	Creates a virtual Ethernet interface and enters interface configuration mode.
Step 4	`Router(config-if)# transport vpls mesh`	Create a full mesh of pseudowires and enters VPLS transport mode.
Step 5	`Router(config-if-transport)# neighbor` *remote-router-id* [**pw-class** pw-class-name]	Specifies the PE routers to be used in the pseudowire.
Step 6	`Router(config-if-transport)# exit`	Exits VPLS transport configuration mode and enters interface configuration mode.
Step 7	`Router(config-if)# switchport`	Configures the port for Layer 2 switching.
Step 8	`Router(config-if)# switchport mode trunk`	Enables permanent trunking mode and negotiates to convert the link into a trunk link.

	Command	Purpose
Step 9	`Router(config-if)# switchport trunk allowed vlan {add | except | none | remove} vlan [,vlan[,vlan[,...]]]`	Configures the list of VLANs allowed on the trunk.
Step 10	`Router(config)# exit`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring an MPLS Traffic Engineering Tunnel

There are several ways to specify the route through which traffic should pass.

- Explicitly specify the PE routers as part of the virtual Ethernet interface configuration
- Configure an MPLS Traffic Engineering tunnel
- Configure a GRE tunnel

The following task explains how to configure an MPLS Traffic Engineering tunnel. For more information about MPLS Traffic Engineering tunnels, see *MPLS Traffic Engineering and Enhancements.*

	Command	Purpose
Step 1	`Router> enable`	Enables privileged EXEC mode (enter your password if prompted).
Step 2	`Router# configure terminal`	Enters global configuration mode.
Step 3	`Router(config)# interface tunnel number`	Configures an interface type and enters interface configuration mode.
Step 4	`Router(config-if)# ip unnumbered type number`	Assigns an IP address to the tunnel interface. An MPLS traffic engineering tunnel interface should be unnumbered because it represents a unidirectional link.
Step 5	`Router(config-if)# tunnel destination ip-address`	Specifies the destination for a tunnel. The *ip-address* keyword is the IP address of the host destination expressed in dotted decimal notation.
Step 6	`Router(config-if)# tunnel mode mpls traffic-eng`	Configures the tunnel encapsulation mode to MPLS traffic engineering.
Step 7	`Router(config-if)# tunnel mpls traffic-eng autoroute announce`	Configures the IGP to use the tunnel in its enhanced SPF calculation.
Step 8	`Router(config-if)# tunnel mpls traffic-eng path-option number {dynamic | explicit {name path-name} | identifier path-number} [lockdown]`	Configures the tunnel to use a named IP explicit path or a path dynamically calculated from the traffic engineering topology database. A dynamic path is used if an explicit path is currently unavailable.
Step 9	`Router(config-if)# exit`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a GRE Tunnel

There are several ways to specify the route through which traffic should pass.

- Explicitly specify the PE routers as part of the virtual Ethernet interface configuration
- Configure an MPLS Traffic Engineering tunnel
- Configure a GRE tunnel

The following task explains how to configure a GRE tunnel. For more information on GRE tunnels, see *Implementing Tunnels*.

	Command	Purpose
Step 1	`Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	`Router# configure terminal`	Enters global configuration mode.
Step 3	`Router(config)# interface type number`	Specifies the interface type and number and enters interface configuration mode. To configure a tunnel, use **tunnel** for the type argument.
Step 4	`Router(config-if)# tunnel mode {gre ip | gre multipoint}`	Specifies the encapsulation protocol to be used in the tunnel.
Step 5	`Router(config-if)# mpls ip`	Enables MPLS on the tunnel.
Step 6	`outer(config-if)# tunnel source {ip-address | interface-type interface-number}`	Configures the tunnel source. • Use the *ip-address* argument to specify the source IP address. • Use the *interface-type* and *interface-number* arguments to specify the interface to use. **Note** The tunnel source and destination IP addresses must be defined on both PE routers.
Step 7	`Router(config-if)# tunnel destination {hostname | ip-address}`	Configures the tunnel destination. • Use the hostname argument to specify the name of the host destination. • Use the ip-address argument to specify the IP address of the host destination. **Note** The tunnel source and destination IP addresses must be defined on both PE routers.
Step 8	`Router(config-if)# exit`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 9	`Router(config)# ip route ip-address tunnel num`	Creates a static route.

These examples show the three supported methods of configuring A-VPLS.

Explicitly Specifying Peer PE Routers

The following example shows how to create two VPLS domains under VLANs 10 and 20. Each VPLS domain includes two pseudowires to peer PE routers 10.2.2.2 and 10.3.3.3. Load-balancing is enabled through the l**oad-balance flow** and **flow-label enable** commands.

```
pseudowire-class cl1
   encap mpls
   load-balance flow
   flow-label enable
!
port-channel load-balance src-mac
!
interface virtual-ethernet 1
   transport vpls mesh
      neighbor 10.2.2.2 pw-class cl1
      neighbor 10.3.3.3 pw-class cl1
   switchport
   switchport mode trunk
   switchport trunk allowed vlan 10, 20
```

Using MPLS Traffic Engineering Tunnels

The following example shows the creation of two VPLS domains and uses MPLS Traffic Engineering tunnels to specify the explicit path.

```
pseudowire-class cl1
   encap mpls
   load-balance flow
   flow-label enable
!
port-channel load-balance src-mac
!
interface Tunnel1
   ip unnumbered Loopback0
   tunnel mode mpls traffic-eng
   tunnel destination 192.168.1.1
   tunnel mpls traffic-eng autoroute announce
   tunnel mpls traffic-eng path-option 1 explicit name LSP1
!
ip explicit-path name LSP1 enable
   next-address 192.168.2.2
   next-address loose 192.168.1.1
!
interface Tunnel2
   ip unnumbered Loopback0
   tunnel mode mpls traffic-eng
   tunnel destination 172.16.1.1
   tunnel mpls traffic-eng autoroute announce
   tunnel mpls traffic-eng path-option 1 explicit name LSP2
!
ip explicit-path name LSP2 enable
   next-address 172.16.2.2
   next-address loose 172.16.1.1
!
interface virtual-ethernet 1
    transport vpls mesh
      neighbor 10.2.2.2 pw-class cl1
      neighbor 10.3.3.3 pw-class cl1
   switchport
   switchport mode trunk
   switchport trunk allowed vlan 10,20
```

Using MPLS over GRE Tunnels

The following example shows the creation of two VPLS domains under VLANs 10 and 20. Each VPLS domain includes two pseudowires to peer PEs 10.2.2.2 and 10.3.3.3. The pseudowires are MPLS over GRE tunnels because the core is IP.

```
pseudowire-class cl1
   encap mpls
   load-balance flow
!
port-channel load-balance src-mac
!
interface tunnel 1
   tunnel mode gre ip
   mpls ip
   tunnel source 10.1.1.1
   tunnel destination 10.2.2.2
!
interface tunnel 2
   tunnel mode gre ip
   mpls ip
   tunnel source 10.1.1.1
   tunnel destination 10.3.3.3
!
interface virtual-ethernet 1
   transport vpls mesh
      neighbor 10.2.2.2 pw-class cl1
      neighbor 10.3.3.3 pw-class cl1
   switchport
   switchport mode trunk
   switchport trunk allowed vlan 10, 20

ip route 10.2.2.2 255.255.255.255 Tunnel1
ip route 10.3.3.3 255.255.255.255 Tunnel2
```

Routed Pseudo-Wire (RPW) and Routed VPLS

RPW and Routed VPLS can route Layer 3 traffic as well as switch Layer 2 frames for pseudowire connections between provider edge (PE) devices. Both point-to-point PE connections, in the form of Ethernet over MPLS (EoMPLS), and Virtual Private LAN Services (VPLS) multipoint PE connections are supported. The ability to route frames to and from these interfaces supports termination of a pseudowire into a Layer 3 network (VPN or global) on the same switch, or to tunnel Layer 3 frames over a Layer 2 tunnel (EoMPLS or VPLS). The feature supports faster network convergence in the event of a physical interface or device failure through the MPLS Traffic Engineering (MPLS-TE) and Fast Reroute (FRR) features. In particular, the feature enables MPLS TE-FRR protection for Layer 3 multicast over a VPLS domain.

Note When the RPW is configured in A-VPLS mode, TE/FRR is not supported because A-VPLS runs over ECMP and the ECMP convergence is comparable to TE/FRR.

To configure routing support for the pseudowire, configure an IP address and other Layer 3 features for the Layer 3 domain (VPN or global) in the virtual LAN (VLAN) interface configuration. The following example assigns the IP address 10.10.10.1 to the VLAN 100 interface, and enables Multicast PIM. (Layer 2 forwarding is defined by the VFI VFI100.)

```
interface vlan 100
  xconnect vfi VFI100
  ip address 10.10.10.1 255.255.255.0
  ip pim sparse-mode
```

The following example assigns an IP address 20.20.20.1 of the VPN domain VFI200. (Layer 2 forwarding is defined by the VFI VFI200.)

```
interface vlan 200
  xconnect vfi VFI200
  ip vrf forwarding VFI200
  ip address 20.20.20.1 255.255.255.0
```