ERICSSON

Configuring Bridging

SYSTEM ADMINISTRATOR GUIDE

7/1543-CRA 119 1170/1-V1 Uen J

Contents

1 Overview

This document describes how to configure the following bridging functions: MAC moves loop detection, Rapid Spanning Tree Protocol (RSTP), bridged interfaces, bridged Ethernet ports, bridged 802.1Q PVCs, bridged ATM PVCs, bridged link groups, bridge subscriber circuits, and bridge virtual interface (BVI) ports.

- The bridging feature in the SmartEdge router implements transparent, self-learning bridges as described in IEEE 802.1D.
- Bridges and bridged interfaces are context specific.
- Multiple bridges can be created within a context.
- Bridged Virtual Interface (BVI) ports are pseudocircuits that support bridging and routing in the same SmartEdge router context to allow a one-armed router function for gateway devices and eliminate the need for additional I/O ports on the router for physical loopback cables.

  BVI ports are created by inserting an IP interface with a virtual MAC into a bridge group to allow any bridging traffic sent to the BVI port to be routed and any IP traffic sent to the BVI port to be bridged. In typical BVI port implementations, a virtual IP interface is assigned to each bridge group that is used to route packets between the bridged and routed networks. This section explains how BVI ports can be used to support different types of networks. For example, you can configure BVI ports to terminate multiple types of Layer 2 circuits into a single bridge group, remove Ethernet switches to media gateways in a mobile-packet-based network, and eliminate physical loopback cables.
- A bridge circuit can be designated as *restricted*:
  - No MAC address learning is allowed.
  - The MAC address of a restricted bridge circuit must be statically configured.
  - Either trunk or tributary circuits can be designated restricted.

This document applies to both the Ericsson SmartEdge® and SM family routers. However, the software that applies to the SM family of systems is a subset of the SmartEdge OS; some of the functionality described in this document may not apply to SM family routers.

For information specific to the SM family chassis, including line cards, refer to the SM family chassis documentation.

For specific information about the differences between the SmartEdge and SM family routers, refer to the Technical Product Description *SM Family of Systems*

(part number 5/221 02-CRA 119 1170/1) in the **Product Overview** folder of this Customer Product Information library.

1.1 BVI Port Operation

Figure 1 shows a BVI port connected to a standard Layer 2 bridging domain, called a bridge group in the operating system. In this case, a need exists to terminate multiple types of Layer 2 circuits into a single bridge group so that local or non-routable traffic is bridged among the bridged interfaces in the same bridge group, while traffic sent to the BVI port is routed.

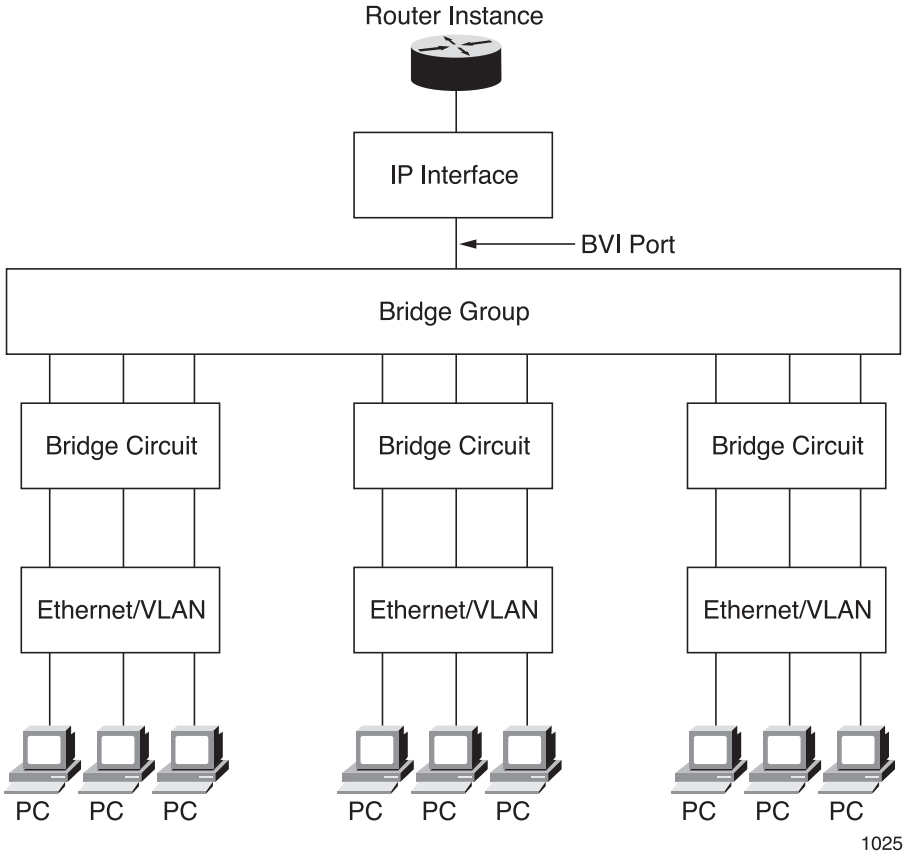

Figure 1 BVI Port Wireline Configuration

To illustrate how BVI ports can bridge local or non-routable packets and forward the rest of the packets as Layer 3 traffic, Figure 2 illustrates a simplified version of the network configuration traffic shown in Figure 1.

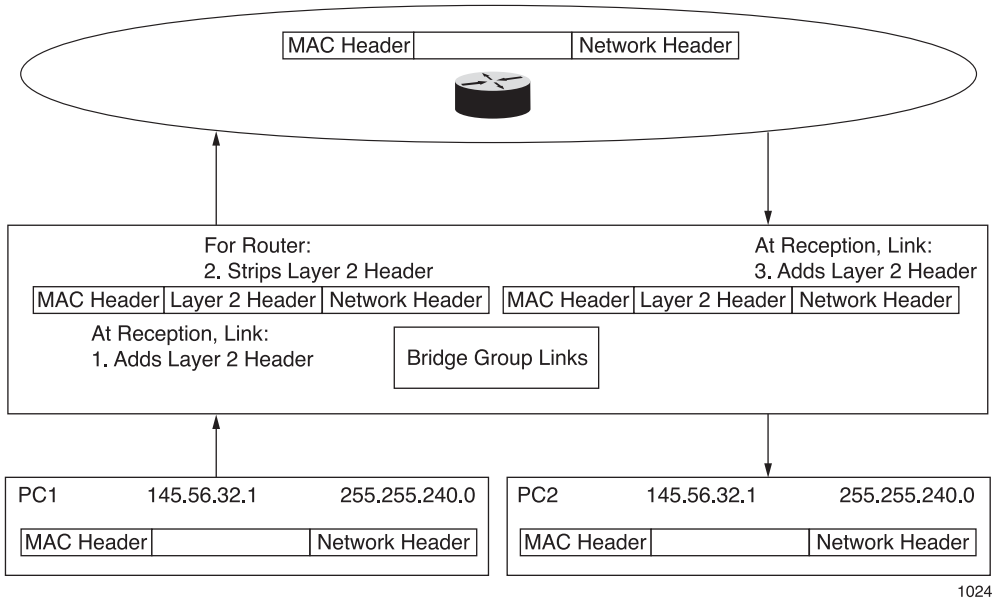

Figure 2 BVI Port Basic Configuration

The simplified BVI port configuration in Figure 2 shows the three layers of headers that are present when traffic goes through the links from PC 1 to PC 2:

1 When traffic goes to the bridge group links, Layer 2 headers are added by the sender.

2 The Layer 2 headers are stripped at the router.

3 The router is configured for bridging IP traffic, which allows the MAC header to be forwarded. If the MAC address of the packet does not match the BVI port MAC, then the traffic is bridged; and if the MAC address of the packet matches the BVI port MAC, then the traffic is routed.

4 New Layer 2 headers are added after traffic exits the router at the next link. Link groups can be attached to different VLANs that have the same Layer 2 header because the traffic is forwarded by the IP destination field (not the Layer 2 header).

1.2 BPDU Queuing

The SmartEdge router can map incoming BPDUs on Ethernet ports, 802.1Q PVCs, and 802.1Q PVCs within 802.1Q tunnels to internal prioritized queues. This mapping takes precedence over other circuit marking.

This feature applies to both IEEE BPDUs and Cisco proprietary Per VLAN Spanning Tree Plus (PVST+) BPDUs.

You can apply BPDU queuing to the following:

- Bridged untagged Ethernet ports

- Bridged 802.1Q PVCs
- Bridged 802.1Q PVCs within 802.1Q tunnels

The following restrictions apply:

- BPDU queuing does not operate on VPLS PW circuits.
- Do not apply the **`bpdu priority`** command to circuits and ports of bridges enabled for RSTP.

This **`bpdu priority`** command sets the priority of the incoming Spanning Tree BPDUs. The queuing of incoming packets is determined by their assigned priority. The range of values for the priority argument is 0 to 7, where 0 is the highest priority and 7 is the lowest priority.

1.3 Rate Limiting Using a Bridge Profile

By default, all inbound bridge traffic is limited by the rate and burst size imposed by the configuration of the port or circuit to which you assign the bridge profile. However, you can apply rate limiting for certain classes of traffic on the bridge, using the bridge profile:

- Broadcast traffic
- Multicast traffic
- Traffic with unknown destination frames

For each traffic class, you can specify a maximum rate and burst size. The system accepts packets of a bridge traffic class that conform to that traffic class rate and burst size without further action; it drops packets that do not conform. See Figure 3.

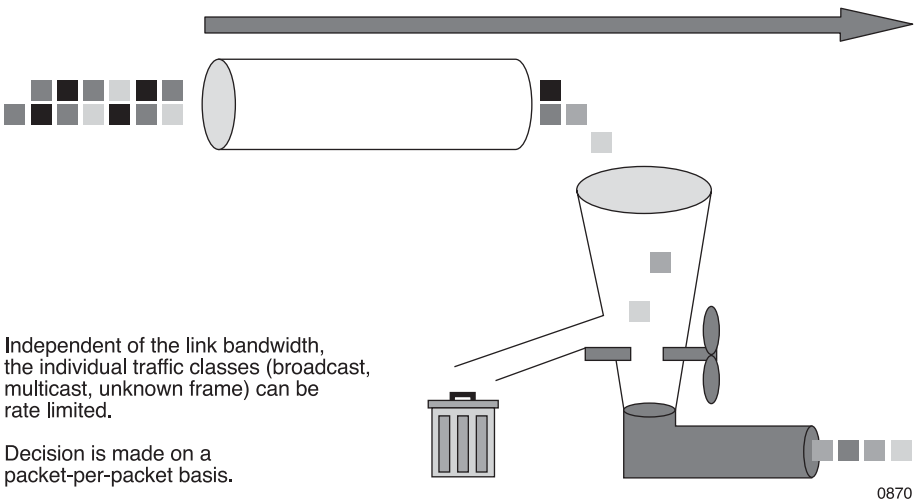

Figure 3 Bridge Profile Rate Limiting

Note: If a quality of service (QoS) policy (or policies) is attached to the port or circuit to which the bridge profile is assigned, and that QoS policy includes rate limiting, that QoS rate limiting is applied to the traffic on the port or circuit after the broadcast, multicast, or unknown frame rate limiting. Packets dropped by the broadcast, multicast, or unknown frame rate limiting are not applied against the QoS rate limiting policy.

1.4 Bridges Requirements and Restrictions

The requirements and restrictions of the SmartEdge router bridges are described in the following sections:

1.4.1 Bridging Supported Circuits

Warning!

Can result in system failure. Do not configure an L2 tunnel on an interface bound to a BVI port. BVI ports are compatible only with L3 forwarding and L3 tunnels.

- Circuits that can be bridged include Ethernet ports with 802.1D or 802.1Q encapsulation, 802.1Q permanent virtual circuits (PVCs), and Asynchronous Transfer Mode (ATM) PVCs with RFC 1483 bridged encapsulation. Packets going out with a 1483 bridged encapsulation type that are configured in a bridge group retain the 802.1Q header tags.
- IP over Ethernet (IPoE) and PPP over Ethernet (PPPoE)-encapsulated circuits can be bridged (at the medium access control (MAC) layer); however, IP- and Point-to-Point Protocol (PPP)-encapsulated circuits cannot be bridged.
- Bridges support both tributary and trunk circuits in any combination:
  - Tributary circuits are access-facing (face subscribers), and trunk circuits are network-facing (face service providers).
  - Packets can be forwarded from:
    - Trunk to trunk
    - Trunk to tributary
    - Tributary to trunk
  - A packet arriving on a tributary circuit with a destination address of another tributary circuit is flooded on all trunk circuits.

– Flooding (unknown destination MAC address or multicast MAC address:

  • Packets arriving on a trunk circuit are flooded to all trunk and tributary circuits.

  • Packets arriving on a tributary circuit are flooded only to all trunk circuits.

1.4.2 RSTP Requirements and Restrictions

The SmartEdge router implements the Rapid Spanning Tree Protocol (RSTP) and MAC moves monitoring to provide path redundancy and to prevent bridging loops. Additional information on RSTP is found in IEEE 802.1d and IEEE 802.1w.

- BVI with RSTP is supported.
- RSTP is not supported over ATM.
- RSTP is supported on VPLS pseudowires (PWs).
- RSTP is supported on bridged link group circuits. (IEEE 802.3ad link aggregation group (LAG) circuits. These include the 802.1Q, Ethernet, and access type link groups.
- Meshed network configurations and other network configurations with multiple bridges running RSTP can be configured in a master-client *RSTP tracking* relationship so that the bridged circuits of the client bridges replicate the blocking, forwarding, and flushing actions of the master bridge. Configurations where RSTP tracking is useful include (1) customer-edge (CE) dual-homed redundant topologies, (2) meshed bridge networks providing multiple access circuits to a client, and (3) bridged networks that require interoperability with Cisco switches.

  – The RSTP tracking master must be an RSTP-enabled bridge.

  – The RSTP tracking clients must be non-RSTP bridges.

  – You cannot configure a bridge as the RSTP master (`master` command) if the bridge is bound to the interface of a C-VLAN in a 802.1Q tunnel (S-VLAN) where the C-VLAN is a circuit in an economical access link group. You can configure a bridge as the STP master when the bridge is bound to the interface of an S-VLAN where the S-VLAN is a circuit in an economical access link group, but only if the economical link group is configured with the `replicate` keyword.

  – You cannot bind the interface of an RSTP client bridge to directly to an economical access link group (LG) if the link group packets are Dot1Q encapsulated; instead, you must bind to the 802.1Q or Q-in-Q PVCs in economical link group. The following configuration excerpt (below) illustrates this idea:

```
link-group lg-eco access economical
 encapsulation dot1q                          //NOTE 1
 dot1q pvc 100
  bind interface client_bridge local          //NOTE 2
```

NOTE 1: Do not bind an economical link-group to a bridge interface'

NOTE 2: Bind to circuits in the economical link group.

- When interworking with a Cisco switch, the Cisco switch cannot be made the bridge root.

1.4.3 BPDU Filtering Requirements and Restrictions

The SmartEdge router can filter bridge protocol data units (BPDUs) on Ethernet ports, 802.1Q PVCs, and 802.1Q PVCs in 802.1Q tunnels. The `bpdu` command controls BPDU filtering

BPDUs can be filtered in either of the following ways:

- All BPDUs are dropped.
- Only BPDUs are allowed.

You can apply BPDU filtering to the following:

- Bridged untagged Ethernet ports
- Bridged 802.1Q PVCs
- Bridged 802.1Q PVCs within 802.1Q tunnels

The following restrictions apply to BDPU filtering:

- BPDU filtering does not operate on Virtual Private LAN Services (VPLS) pseudowires (PW) circuits.
- Do not apply the `bpdu` command to circuits and ports of bridges enabled for Rapid Spanning Tree Protocol (RSTP).

2 Configuration and Operations Tasks

This section describes how to configure bridging and perform operation tasks on bridges.

2.1 Configuration Guidelines

The following guidelines apply when configuring bridging:

- Bridges are context specific. You can configure multiple bridges in each context, but you cannot forward traffic from a circuit associated with one bridge to another bridge, either within the same context or in a different context.

  You can use physical cabling for forwarding traffic from one bridge to another.

- When creating a bridge for 802.1Q PVCs, you must explicitly propagate the 802.1Q priority from ingress to egress by using the **`propagate-qos-from-ethernet`** and **`propagate-qos-to-ethernet`** commands (in dot1q profile configuration mode).

- To associate one or more bridged circuits with a bridge, you must configure a bridged interface and associate with it the name of an existing bridge; bridged circuits are associated with the named bridge by being bound to the same bridged interface.

  Caution!

  Risk of data loss. Inbound packets can be dropped without warning if the maximum transmission unit (MTU) of the port with the outbound circuit is not as large as the MTU of the port with the inbound circuit. To reduce the risk, always configure every port with circuits bound to a bridged interface with the same MTU value.

- A subscriber record can contain either a bridge or an IP address, but not both.

- When creating a VPLS-enabled bridge with one or more embedded VPLS pseudowires, you must specify the pseudowire IDs and the IP addresses of the VPLS neighbors. See the *Configuring VPLS* document for more detailed information on VPLS-enabled bridges.

2.2 Bridging Step-by-Step Configuration Procedures

To configure bridging, perform the tasks described in the following sections.

2.2.1 Create a Named Bridge

To enable bridging in a context, create a named bridge by performing the tasks described in Table 1. More than one bridge can be created in any context.

Table 1 Configure the Bridge

	Task	**Root Command**	**Notes**
1.	Create a bridge and access bridge configuration mode.	*bridge*	Enter this command in context configuration mode.
2.	Specify bridge attributes. All of the following commands are in bridge configuration mode:		
2. cont'd	Specify the aging time for inactive learned MAC addresses, after which they are dropped.	*aging-time*	
2. cont'd	Set an alias for the bridge MAC address	*bridge-mac-address*	
2. cont'd	Associate a description with the bridge.	*description (bridge)*	
2. cont'd	Configure bridge IGMP attributes.	*igmp snooping*	
2. cont'd	Enable or disable the learning of MAC addresses.	*learning*	Learning is enabled by default.
2. cont'd	Enable detection of bridging loops based on counting MAC moves within the bridge. Enter the configuration of loop-detection parameters of the bridge	*loop-detection*	See Configure Mac Moves Loop Detection Attributes in a Bridge.
2. cont'd	Specify one or more MAC addresses for packets drop.	*mac-entry*	Enter this command for each MAC address that is not allowed on this bridge.
2. cont'd	When loop-detection has blocked a circuit 5 times, block the MAC address observed with the loop so that it is not allowed on the bridge.	*mac-move-drop*	To disable this behavior, enter **`no mac-move-drop`**.

Table 1 Configure the Bridge

	Task	Root Command	Notes
2. cont 'd	Enable the bridge for the VPLS and enter VPLS bridge configuration mode.	*vpls*	See the *Configuring a VPLS-Enabled Bridge* section in the *Configuring VPLS* document for information on the commands in VPLS bridge configuration mode.
2. cont 'd	Enable or disable VPLS	*disable (VPLS)*	Enter in VPLS bridge configuration mode.
2. cont 'd	Specify the VPLS profile that defines the bridged VPLS neighbors.	*profile (VPLS)*	
2. cont 'd	Specify the VPLS pseudowire.	*pw-id*	
2. cont 'd	Enable RSTP for the bridge.	*spanning-tree (bridge)*	See Section 2.2.3 on page 13..
2. cont 'd	Configure the current bridge to track the RSTP master bridge.	*track spanning-tree*	When a VPLS-enabled bridge is configured for tracking, the pseudowire must follow the same LSP hops as the master. You can enable tracking only for bridges not running RSTP. See `master` command in Section 2.2.3 on page 13. See the RSTP tracking requirements and restrictions listed in Section 1.4 on page 5.

2.2.2 Configure MAC Moves Loop Detection

MAC moves loop detection breaks bridging loops by monitoring the rates at which MAC moves are occurring. If the rate exceeds the configured threshold, an appropriate circuit in the bridge (or MAC address) is blocked to break the loop. The following ports and circuits can be configured for loop detection using this method:

- Virtual Private LAN Services (VPLS) pseudowires
- Ethernet ports

- 802.1Q (dot1q) PVCs
- Access link groups, 802.1Q link groups, and Ethernet link groups

To configure MAC moves loop detection:

1 Configure the bridge (Table 1) with the `loop-detection` command, or accept the static loop-detection default as described in the `mac-move-drop` command. By default, MAC move dropping is configured for bridges, but if you enable dynamic loop detection with the `loop-detection` command, static loop detection is automatically disabled. To disable both forms of loop detection, enter only the `no mac-move-drop` command.

2 If you enabled dynamic loop detection by entering the `loop detection` command in the previous step, you can accept the default settings, or change their settings as show in Table 2.

3 If you enabled dynamic loop detection, you need to set the per-port, per-circuit, per-link group or per-pseudowire loop detection priority, or accept the default priority. See Table 6 for the configuration of `priority (loop-detection)` in the bridge profile. (The bridge profile is applied to bridging-capable ports, circuits, link groups or pseudowires.

Table 2 Configure MAC Moves Loop Detection Attributes of a Bridge

	Task	**Root Command**	**Notes**
1.	Enable detection of bridging loops based on counting MAC moves within the bridge. Enter the configuration of loop-detection parameters of the bridge.	*loop-detection*	Enter this command in bridge configuration mode.
2.	Specify the following loop-detection attributes. All of the following commands are in loop-detection configuration mode:		
	Configure the amount of time over which MAC moves frequency is averaged.	*interval*	loop-detection mode
	Set the initial time a circuit remains blocked after a bridging loop is detected.	*block-time*	loop-detection mode
	Set the threshold above which a bridging loop is declared.	*move-frequency*	loop-detection mode

Table 3 Configure MAC Moves Loop Detection Attributes of a Port

	Task	Root Command	Notes
1.	Enter loop-detection configuration mode to set the optional loop-detection attributes.	*loop-detection*	Enter this command in bridge configuration mode. This command enables loop detection on the bridge. Detection of bridging loops is based on counting MAC moves within the bridge.
2.	Specify the following loop-detection attributes. All of the following commands are in loop-detection configuration mode:		
	Configure the amount of time over which MAC moves frequency is averaged.	*interval*	loop-detection mode
	Set the initial time a circuit remains blocked after a bridging loop is detected.	*block-time*	loop-detection mode
	Set the threshold above which a bridging loop is declared.	*move-frequency*	loop-detection mode

2.2.3 Configure RSTP Attributes in a Bridge

Table 4 shows how to configure a bridge for RSTP. Enter all commands in spanning-tree configuration mode, unless otherwise noted.

Table 4 Configure RSTP Attributes

	Task	Root Command	Notes
1.	Access spanning-tree configuration mode for the current bridge in the current context.	*spanning-tree (bridge)*	Enter this command in bridge configuration mode. RSTP is not supported over ATM.

Table 4 Configure RSTP Attributes

	Task	Root Command	Notes
2.	Specify RSTP attributes. All of the following commands are in spanning-tree configuration mode:		
	Set the maximum allowed average rate and burst rate of received bridge protocol data units (BPDUs).	*bpdu rate-limit*	
	Set forward delay time.	*forward-delay*	
	Set group MAC address (destination address field in BPDUs).	*group-mac-address*	The source address is the MAC address of the SmartEdge router controller card.
	Set the interval between sending BPDUs (Spanning Tree Protocol (STP) Hellos).	*hello-interval (spanning-tree)*	
	Specify whether the current bridge is an RSTP master.	*master*	Bridges that are not running RSTP and are enabled for tracking by the `track spanning-tree` command (Table 1) are called *client* bridges. When the state of the circuit controlled by the master bridge changes to blocking, forwarding, or flushing, all circuits on the same port of the tracking client bridges change to the same state.
	Set the maximum age of received BPDUs.	*max-age*	
	Set the bridge priority.	*priority (spanning-tree)*	
	Control the STP process on the bridge.	no *shutdown (Spanning Tree)*	The default state is no shutdown.
	Set transmit hold count.	*transmit-hold count*	

2.2.4 Configure a Bridged Interface

To enable bridging on an interface and to specify the interface used by the bridge, perform the tasks described in Table 5.

Table 5 Configure a Bridged Interface

	Task	Root Command	Notes
1.	Create a bridged interface and access the interface configuration mode.	*interface (context)*	Enter this command in context configuration mode. Specify the **`bridge`** keyword. For more information about this command and other interface attributes, see *Configuring Contexts and Interfaces*.
2.	Associate it with a bridge.	*bridge* **`name`** ***`bridge-name`***	Enter this command with the keyword `bridge` followed by the name of a bridge in the current context.

2.2.5 Configure Bridge Profiles

Table 6 describes how to create a bridge profile. Bridge profiles are assigned to ports, circuits, and link groups that are bound to bridged interfaces. Bridge profiles are also assigned to subscribers associated with bridges.

Note: You can assign bridge profiles to subscribers and VPLS neighbors associated with bridges. When the subscriber circuit is bound to a bridged interface, the attribute values in the named bridge profile assigned to the VPLS neighbor override those in the default bridge profile for the circuit, unless the circuit is also assigned a named bridge profile.

Enter all commands of Table 6 in bridge profile configuration mode, unless otherwise noted.

Table 6 Configure a Bridge Profile

	Task	Root Command	Notes
1.	Create a named or default bridge profile and access bridge profile configuration mode.	*bridge profile*	Enter this command in global configuration mode.
2.	Specify bridge profile attributes (step 3 through step 13). Unless otherwise specified enter these commands in bridge profile configuration mode.		
3.	Sets the priority of the incoming BPDUs on the port or circuit to which you assign this bridge profile.	*bpdu priority*	Do not apply the **`bpdu priority`** command to circuits and ports of bridges enabled for RSTP.

Table 6 Configure a Bridge Profile

	Task	Root Command	Notes
4.	Specify the filtering of received BPDUs on the port or circuit to which you assign this bridge profile.	*bpdu*	The **bpdu** command options follow: • **bpdu deny** — Drop all received BPDUs, and pass all other received traffic. • **no bpdu deny** — Pass all received BPDUs. • **bpdu allow-only** — Pass all received BPDUs and drop all other received traffic. • **no bpdu allow-only** — Pass all non-BPDU received traffic. Do not apply the **bpdu** command to circuits and ports of bridges enabled for RSTP.
5.	Create a filter that drops incoming packets when their source MAC address matches any entry in the MAC list. This command also enters the mac-list configuration mode in which MAC addresses in this list are specified.	*mac-list*	Enter this command in global configuration mode.
6.	Include the MAC-list filter criteria in the current bridge profile.	*drop source*	The MAC-list is used as a filter that causes incoming packets with source MAC addresses matching the list to be dropped.
7.	Specify the type of bridged circuit as a trunk.	*trunk*	The default type is tributary. If a bridge profile is associated with a VPLS neighbor, the bridge circuit must be trunk type.
8.	Specify whether MAC addresses are restricted for the port, circuit, or VPLS pseudowire circuit to which you assign this bridge profile.	*restricted*	The default value is unrestricted.
9.	Specify the maximum number of dynamic MAC addresses for the port, circuit, or VPLS pseudowire circuit to which you assign this bridge profile.	*mac-limit*	The default value is unlimited if the circuit type is trunk; the default value is 4 if the circuit type is tributary.

Table 6 Configure a Bridge Profile

	Task	Root Command	Notes
10.	Set the rate and burst tolerance for broadcast traffic on any port, circuit, or VPLS pseudowire circuit to which you assign this bridge profile.	*broadcast rate-limit*	
11.	Set the rate and burst tolerance for multicast traffic on any port, circuit, or VPLS pseudowire circuit to which you assign this bridge profile.	*multicast rate-limit*	
12.	Set the rate and burst tolerance for traffic to unknown destinations on any port, circuit, or VPLS pseudowire circuit to which you assign this bridge profile.	*unknown-dest rate-limit*	
13.	Enter the loop-detection profile configuration mode.	*loop-detection*	
	Set the priority of loop-prevention blocking on the ports and circuits assigned to this bridge profile.	*priority (loop-detection)*	This command is entered in the loop-detection profile configuration mode.

2.2.6 Configure a Spanning-Tree Profile

Table 7 describes how to create a spanning-tree profile. Spanning-tree profiles are assigned to ports and circuits that are bound to bridged interfaces (see tables).

Enter all commands of Table 7 in spanning-tree profile configuration mode, unless otherwise noted.

Table 7 Configure RSTP Profile

	Task	Root Command	Notes
1.	Create a spanning-tree profile and enter the spanning-tree profile configuration mode.	*spanning-tree profile*	Enter this command in the global configuration mode.

Table 7 Configure RSTP Profile

	Task	Root Command	Notes
2.	Specify spanning-tree profile attributes:		
	Set the Rapid Spanning Tree Protocol (RSTP) cost of the assigned port.	*cost (spanning-tree)*	Also called the Spanning Tree Protocol "port cost."
3.	Set the associated port as a Rapid Spanning Tree Protocol (RSTP) edge port.	*edge-port*	
4.	Enable sending Bridge Protocol Data Units (BPDUs) to the group MAC address.	*l2protocol-tunnel*	If Layer 2 Protocol tunnel is enabled, the SmartEdge router can send BPDUs directly to external customer edge (equipment) and allow the SmartEdge router to participate in the customer's spanning-tree domain.
5.	Treat the associated port as always connected to a point-to-point link.	*p2p-port*	
6.	Set the spanning-tree priority of the assigned port.	*port-priority*	

2.2.7 Configure a Bridged Ethernet Port

To configure a bridged Ethernet port, perform the tasks described in Table 8.

Table 8 Configure a Bridged Ethernet Port

	Task	Root Command	Notes
1.	Select the Ethernet port and enter port configuration mode.	*port ethernet*	Enter this command in global configuration mode.
2.	Assign a bridge profile.	*bridge profile*	The default bridge profile is assigned automatically if you do not enter this command.
3.	Specify bridge attributes for the port:		
	Specify the static MAC addresses.	*bridge mac-entry*	Enter this command for the MAC address of each station known to be on this port. The bridge dynamically learns the addresses of other stations as they connect to the port.

Table 8 Configure a Bridged Ethernet Port

	Task	Root Command	Notes
4.	Bind the port to an existing bridged interface in an existing context.	*bind interface*	
5.	Assign a spanning-tree profile to the port.	*spanning-tree profile* ***profile-name***	See Table 7.

Caution!

Risk of data loss. Inbound packets can be dropped without warning if the maximum transmission unit (MTU) of the port with the outbound circuit is not as large as the MTU of the port with the inbound circuit. To reduce the risk, always configure every port with circuits bound to a bridged interface with the same MTU value.

Note: Configuration commands for other port attributes are not included in Table 8. For information about configuring Ethernet ports, see *Configuring ATM, Ethernet, and POS Ports*.

2.2.8 Configure a Bridged 802.1Q PVC

To configure a bridged 802.1Q PVC, perform the tasks described in Table 9.

Table 9 Configure a Bridged 802.1Q PVC

	Task	Root Command	Notes
1.	Select the Ethernet port and access port configuration mode.	*port ethernet*	Enter this command in global configuration mode.
2.	Specify 802.1Q encapsulation for the Ethernet port.	*encapsulation (Ethernet port)*	Set encapsulation to `dot1q`.
4.	Create either a standard 802.1Q PVC or transport-enabled 802.1Q PVC and configure the dot1q PVC options. Enter command in port configuration mode.		
		dot1q pvc	For standard 802.1Q PVCs, use this command.
		dot1q pvc transport	For transport-enabled 802.1Q PVCs, use this command.

Table 9 Configure a Bridged 802.1Q PVC

	Task	Root Command	Notes
5.	Propagate Ethernet 802.1p user priority bits to IP Differentiated Services Code Point (DSCP) bits.	*propagate qos from ethernet*	Enter these commands in dot1q profile configuration mode. For more information, see *Configuring Circuits for QoS*.
6.	Propagate IP Differentiated Services Code Point (DSCP) bits to Ethernet 802.1p user priority bits.	*propagate qos to ethernet*	
7.	Assign a bridge profile.	*bridge profile*	Enter this command in dot1q PCV configuration mode.
8.	Specify the static MAC addresses.	*bridge mac-entry*	Enter this command for the MAC address of each station known to be on this PVC. The bridge dynamically learns the addresses of other stations as they connect to the PVC.
9.	Bind the circuit to an existing bridged interface with one of the following tasks:		
9. cont'd	Create a static binding to an interface.	*bind interface*	Enter this command in dot1q PVC configuration mode.
9. cont'd	Create a static binding through a subscriber record to an interface.	*bind subscriber*	Enter this command in dot1q PVC configuration mode.
10.	Assign a spanning-tree profile.	*spanning-tree profile*	See Table 7.

Caution!

Risk of data loss. Inbound packets can be dropped without warning if the maximum transmission unit (MTU) of the port with the outbound circuit is not as large as the MTU of the port with the inbound circuit. To reduce the risk, always configure every port with circuits bound to a bridged interface with the same MTU value.

Note: Configuration commands for other 802.1Q circuit attributes are not included in Table 9. For information about configuring 802.1Q PVCs, see *Configuring Circuits*.

2.2.9 Configure a Bridged ATM PVC

To configure a bridged ATM PVC, perform the tasks described in Table 10.

Table 10 Configure a Bridged ATM PVC

	Task	Root Command	Notes
1.	Select the ATM port and access ATM OC configuration mode.	*port atm*	Enter this command in global configuration mode.
3.	Create the ATM PVC and access ATM PVC configuration mode.	*atm pvc*	Specify the **`bridge1483`** keyword for the encapsulation.
4.	Assign a bridge profile.	*bridge profile*	
5.	Specify the static MAC addresses.	*bridge mac-entry*	Enter this command for the MAC address of each station on this PVC. The bridge dynamically learns the addresses of other stations as they connect to the PVC.
6.	Bind the ATM PVC to an existing bridged interface.	*bind interface*	

Caution!

Risk of data loss. Inbound packets can be dropped without warning if the maximum transmission unit (MTU) of the port with the outbound circuit is not as large as the MTU of the port with the inbound circuit. To reduce the risk, always configure every port with circuits bound to a bridged interface with the same MTU value.

Note: Configuration commands for other ATM PVC attributes are not included in Table 10. For information about configuring ATM PVCs, see *Configuring Circuits*.

2.2.10 Configure a Bridged Link Group

You can create 802.1Q, Ethernet, or access link groups that aggregate bridge interfaces. Although the configuration of these three link groups differ, they all share the tasks described in Table 11 relevant to bridge interfaces.

Table 11 Configure a Bridged Link Group

	Task	Root Command	Notes
1.	Select the link group and enter link group configuration mode.	*link-group (Global, DS-1, E1, Port Configuration Modes)*	Enter this command in global configuration mode.
2.	Assign a bridge profile.	*bridge profile*	The default bridge profile is assigned automatically if you do not enter this command.
4.	Specify the static MAC addresses.	*bridge mac-entry*	Enter this command for the MAC address of each station known to be on this link group. The bridge dynamically learns the addresses of other stations as they connect to the link group.
5.	Bind the link group to an existing bridged interface in an existing context.	*bind interface*	
6.	Assign a spanning-tree profile.	*spanning-tree profile*	See Table 7.

The following sections in *Configuring Link Aggregation* describe the full configuration of Ethernet, 802.1Q, and access link groups:

- *Configuring an Ethernet Link Group*
- *Configuring an 802.1Q Link Group*
- *Configuring an Access Link Group*

Caution!

Risk of data loss. Inbound packets can be dropped without warning if the maximum transmission unit (MTU) of the port with the outbound circuit is not as large as the MTU of the port with the inbound circuit. To reduce the risk, always configure every port with circuits bound to a bridged interface with the same MTU value.

Note: Configuration commands for other port attributes are not included in Table 8. For information about configuring Ethernet ports, see *Configuring ATM, Ethernet, and POS Ports*.

2.2.11 Configure a Bridged Subscriber

To configure a subscriber record, named profile, or default profile for bridging, perform the tasks described in Table 12. Also see Section 3.1 on page 31.

Table 12 Configure a Bridged Subscriber

	Task	Root Command	Notes
1.	Create the subscriber record, named profile, or default profile and access subscriber configuration mode.	*subscriber (context configuration)*	Enter this command in context configuration mode. For more information, see *Configuring Subscribers*.
2.	Assign a bridge profile to be used by the circuit on which the subscriber session occurs.	*bridge profile*	
3.	Associate the subscriber with an existing bridge.	*bridge*	

Note: Configuration commands for other subscriber attributes are not included in Table 12. For information about configuring subscribers, see *Configuring Subscribers*.

2.2.12 Configure a BVI Port

To configure a bridged virtual interface port, perform the tasks described in Table 13 and see Bridged Virtual Interface Port: Example. The configured BVI port supports routing on Network Layer 3 and bridging on Link Layer 2 at the same time using one instead of two ports.

Table 13 Configure a Bridged Virtual Interface Port

	Task	Root Command	Notes
This procedure creates a BVI port, binds an interface with an IP address to the port, and configures the port to be part of a bridge. The interface with an IP address and the bridge, and the context they are part of, were created prior to this procedure.			
1.	Create the BVI port.	*port bvi*	Enter this command in global configuration mode and the following commands in port configuration mode.
2.	Control the BVI port.	no *shutdown (Port)*	

Table 13 Configure a Bridged Virtual Interface Port

	Task	Root Command	Notes
3.	Bind the BVI port to an existing IP interface within an existing BVI context.	*bind interface*	Because the interface has an IP address, it supports routing. Do not bind the port to a bridged interface.
4.	Associate the BVI port to be part of an existing bridge that is configured without VPLS bridging attributes.	*bridge name*	The bridge in this step and the interface in the previous step must be in the same context. The bridge cannot be VPLS enabled.

Note: Configuration commands for other attributes are not included in Table 13. For information about binding, see *Configuring Bindings*.

2.2.13 Configure a Bridged VPLS Neighbor

See the section "*Configure a VPLS Profile*" in the *Configuring VPLS* document for the steps used configuring a bridged VPLS neighbor. See the `VPLS` command in Section 2.2.1 on page 10.

2.3 Validating IP Host Connectivity

To discover and troubleshoot the IP address of a CPE, detect duplicate IP addresses, and send ICMP echo requests to the CPE, perform the tasks described in Table 14.

Table 14 Validating IP Host Connectivity

Task	Root Command	Notes
Ping a particular CPE to resolve the MAC address, detect duplicate IP addresses in the system, and test the data path by sending ICMP echo requests to the CPE.	**ping cpe** [***number-of-pings***] ***dest-ip-address source-ip-address*** **bridge** ***bridge-name***[**context** ***context_name***] [***options***]	Include the optional ***number-of-pings*** argument to specify the number of pings to transmit. The default number of pings transmitted is 5. Replace the ***dest-ip-address*** argument with the IP address of the CPE you want to ping. Replace the ***source-ip-address*** argument with an unused IP address from the same subnet as the destination IP address. Replace the ***bridge-name*** argument with the name of the bridge that you want to ping. Include the optional **context** ***context_name*** construct in the **ping cpe** command if you want to ping a bridge that exists in a different context. Include any of the optional constructs, keywords, and arguments to configure various ping options as desired. If the ping is successful, various statistics and configuration information are displayed for the specified CPE.

To validate that ARP is operational and resolve the MAC address of an IP host on a CPE, perform the tasks described in Table 15.

Table 15 Resolving the MAC Address of an IP Host on a CPE

Task	Root Command	Notes
Ping a particular CPE over a bridge by initiating an ARP request from the PE to all access circuits and PWs that are configured on the bridge.	**ping arp** ***dest-ip-address*** **bridge** ***bridge-name*** **[context** ***context_name***] **source-ip-address** [***options***]	Use the **ping arp** command to resolve the MAC address and to detect duplicate IP addresses in the system. Replace the ***dest-ip-address*** argument with the IP address of the CPE you want to ping. Replace the ***source-ip-address*** argument with an unused IP address from the same subnet as the destination IP address. Replace the ***bridge-name*** argument with the name of the bridge that you want to ping. Include the optional **context** ***context_name*** construct in the **ping cpe** command if you want to ping a bridge that exists in a different context. Include any of the optional constructs, keywords, and arguments to configure various ping options as desired. If the ping is successful, various statistics and configuration information about all access circuits and PWs that are configured between the source PE and the destination CPE are displayed.

2.4 Troubleshooting Problems in a Bridging Domain

To troubleshoot problems in a bridging domain, the operating system supports the **ping cpe** and **ping arp** commands, which allow you to perform the following tasks:

- Discover an IP host
- Detect duplicate IP addresses
- Troubleshoot the data path to the host

Use the **ping cpe** command to resolve the MAC address, detect duplicate IP addresses in the system, and test the data path by sending Internet Control Message Protocol (ICMP) echo requests to the customer premises equipment (CPE). After you initiate the **ping cpe** command from your provider edge (PE) router, the following operations occur:

1 An Address Resolution Protocol (ARP) request is broadcast in the Layer 2 (L2) bridging domain.

2 The CPE sends an ARP response back to the PE. The ARP response contains the MAC address of the CPE. The PE uses the CPE MAC address in the ARP response to generate an ICMP echo request message that is sent to the CPE.

   If multiple hosts reply to the ARP request, the ping is aborted, indicating that duplicate IP addresses exist in the system.

3 An ICMP echo reply message is processed by the PE and the round-trip time (RTT) is logged.

Note: The **ping cpe** command does not detect duplicate MAC addresses. To detect duplicate MAC addresses, use the feature described in ***IEEE P802.1ag, Connectivity Fault Management (CFM)***.

The **ping arp** command can also be used to resolve the MAC address and to detect duplicate IP addresses in the system. However, the **ping arp** command does not send ICMP echo requests to the CPE.

Note: We recommend using the **ping arp** and **ping cpe** commands in a pure L2 network in which the CPE is an L3 device. If the L3 device acts as an ARP proxy for a host that is not directly connected to the SmartEdge router, the ping still works but is routed over the customer network and the bridging domain.

To discover and troubleshoot the location of an IP host, you must first determine the PE that is connected to the CPE to which the IP host is connected. Use the **ping mpls mac-address** command to display information about the access circuits on the remote PE that is connected to the CPE. Include the **trace** keyword in the **ping mpls mac-address** command to display information about the path that connects the CPE to the PE in the command output. Before executing the **ping mpls mac-address** command, use the **ping arp** command to ensure that the MAC address of the CPE has been learned by the bridge.

Note: The **ping mpls mac-address** command operates over PWs only. If the IP host is attached to the CPE through other devices, further tracing is not possible.

Consider the following restrictions before using the **ping cpe** or **ping arp** command to troubleshoot problems in a bridging domain:

- The **cpe ping** and **ping arp** commands do not work over transport circuits in a bridge.
- The **cpe ping** and **ping arp** commands do not work if a BVI is configured in a bridge.

2.5 Bridge Monitoring, Troubleshooting, and Operations Management Commands

To monitor, troubleshoot, and manage the operation of bridges, perform the tasks listed in Table 16. Enter the **clear** commands in exec mode; enter the **show** commands in any mode.

Table 16 Bridge Operations Commands

Task	Root Command
Unblock all circuits blocked by MAC moves loop detection and clear all MAC moves loop-detection counters.	*clear bridge loop-detection*
Clear the bridge table for a specified bridge.	*clear bridge table*
Disable MAC moves loop detection on the specified circuit.	*clear circuit loop-detection*
Enable the generation of debug messages for bridge-related events and entities.	*ip arp*
Clear the spanning-tree counters for the bridge instance.	*clear spanning-tree*
Clear the spanning-tree counters for the specified circuits on the bridge.	*clear spanning-tree circuit*
Display bridge profile circuit assignments.	*show bridge associations*
Display bridge binding information. Use the `tracking` keyword to display information about RSTP client master tracking.	*show bridge bindings*
Display information for configured bridges.	*show bridge info*
Display the MAC moves loop-detection status of the specified bridge.	*show bridge loop-detection*
Display information for configured bridge profiles.	*show bridge profile*
Display statistics for one or more bridges.	*show bridge statistics*
Display the bridge forwarding table for one or more bridges.	*show bridge table*
Display the bridge forwarding table for one or more bridges that know the specified MAC address.	*show bridge table mac-entry*

Table 16 Bridge Operations Commands

Task	Root Command
Display the bridge forwarding table for one or more bridges with the specified circuit type.	*show bridge table type*
Display spanning-tree information for the bridge instance.	*show spanning-tree*
Display spanning-tree information for specific circuits on the bridge.	*show spanning-tree circuit*
Display the names of the RSTP master bridge or bridges and the names of their non-RSTP client bridges.	*show spanning tree track*
Display the status of peers linked by VPLS bridge.	*show vpls peer*

3 Configuration Examples

The examples in this section provide partial command samples to illustrate the configuration of bridges. In most examples, only the commands specifically needed for bridging are included.

3.1 Two Basic Bridges: Example

This example, illustrated by Figure 4, creates a context and two bridges with default attributes, except for those configured.

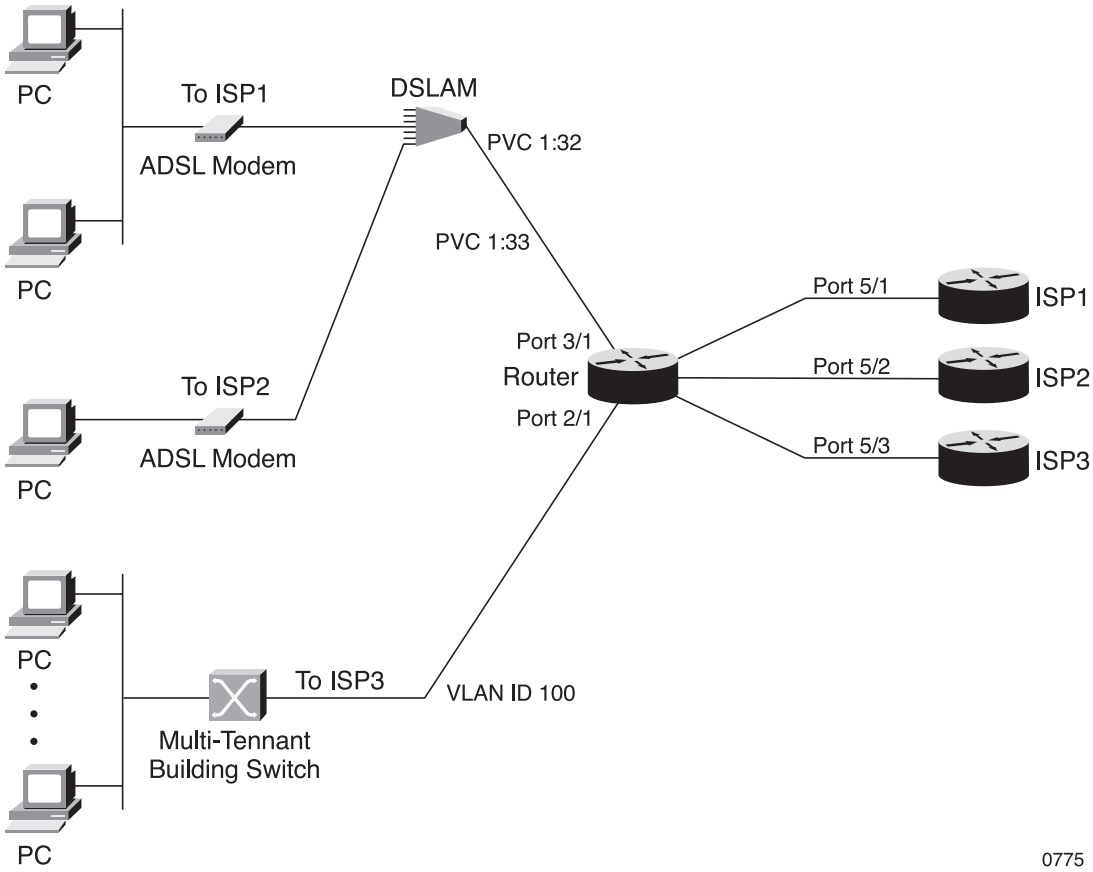

Figure 4 Bridging in a Mixed Environment

Figure 5 shows the logical connections between subscribers on the access network to ISPs through the bridges configured on the SmartEdge router for the configuration shown in Figure 4.

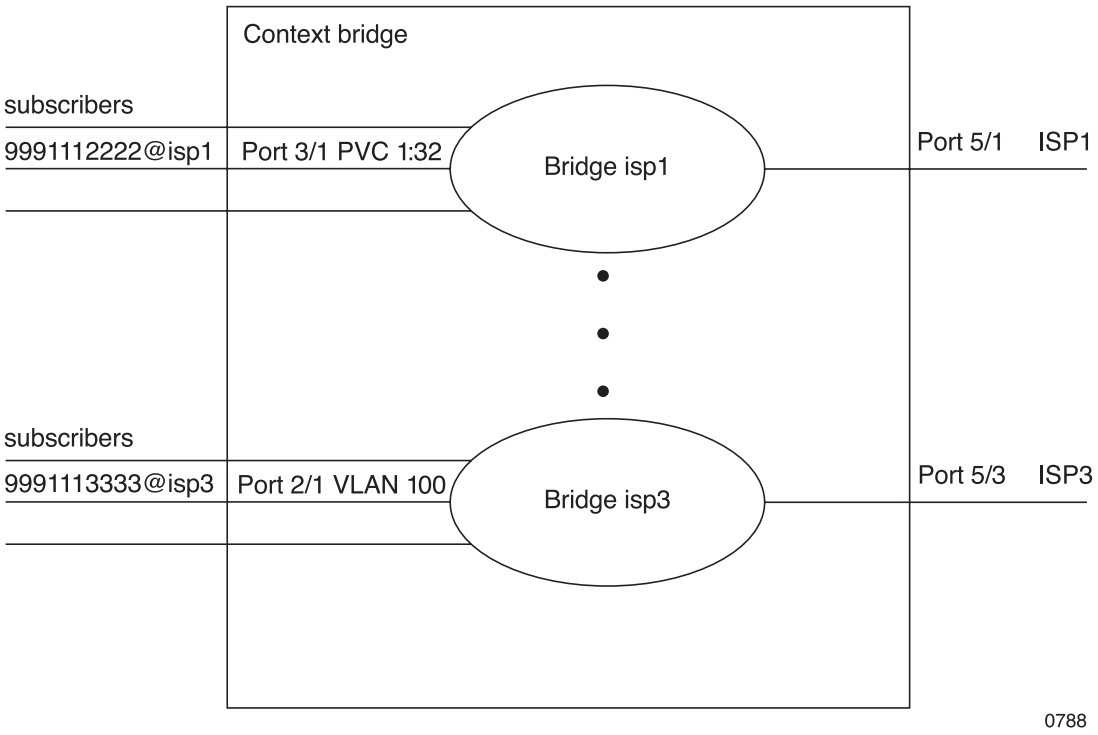

Figure 5 Bridged Subscribers to ISP Connections

In the following command lines, two bridges (`isp1` and `isp3`) are created in the context named `bridge`.

```
[local]Redback(config)#context bridge
[local]Redback(config-ctx)#bridge isp1
[local]Redback(config-bridge)#description Bridge for all traffic to ISP1
[local]Redback(config-bridge)#aging-time 18000
[local]Redback(config-ctx)#bridge isp3
[local]Redback(config-bridge)#description Bridge for all traffic to ISP3
[local]Redback(config-bridge)#aging-time 18000
```

An interface to the isp1 bridge is created and given the name `if-isp1`. Similarly, an interface to the isp2 bridge is created and given the name `if-isp2`.

```
[local]Redback(config)#context bridge
[local]Redback(config-ctx)#interface if-isp1 bridge
[local]Redback(config-if)#bridge name isp1
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#interface if-isp3 bridge
[local]Redback(config-if)#bridge name isp3
```

Two bridge profiles are created in the following command lines. The `prof-isp-trunk` profile is restricted and on a trunk circuit, while the `prof-sub-isp-trib` profile is unrestricted on a tributary circuit:

```
[local]Redback(config-ctx)#bridge profile prof-isp-trunk
[local]Redback(config-bridge-profile)#mac-limit 10
[local]Redback(config-bridge-profile)#restricted
[local]Redback(config-bridge-profile)#trunk
[local]Redback(config-ctx)#bridge profile prof-sub-isp-trib
[local]Redback(config-bridge-profile)#mac-limit 10
[local]Redback(config-bridge-profile)#no trunk
```

The following selects a Gigabit Ethernet port and configures it as a trunk circuit to ISP1:

```
[local]Redback(config)#port ethernet 5/1
[local]Redback(config-port)#bridge profile prof-isp-trunk
[local]Redback(config-port)#mtu 1500
[local]Redback(config-port)#bridge mac-entry 00:d0:ba:04:d8:05
[local]Redback(config-port)#bridge mac-entry 00:0a:0a:04:d8:06
[local]Redback(config-port)#bind interface if-isp1
```

The following selects a Gigabit Ethernet port and configures it as a trunk circuit to ISP3:

```
[local]Redback(config)#port ethernet 5/3
[local]Redback(config-port)#bridge profile prof-isp-trunk
[local]Redback(config-port)#mtu 1500
[local]Redback(config-port)#bridge mac-entry 00:d0:ba:04:d8:07
[local]Redback(config-port)#bridge mac-entry 00:0a:0a:04:d8:08
[local]Redback(config-port)#bind interface if-isp3
```

The following selects an ATM OC port, configures it with an ATM PVC, and configures the PVC as a tributary circuit for ISP1 subscribers:

```
[local]Redback(config)#port atm 3/1
[local]Redback(config-port)#mtu 1500
[local]Redback(config-atm-oc)#atm pvc 1 32 profile ubr encapsulation bridge1483
[local]Redback(config-atm-pvc)#bridge profile prof-sub-isp1-trib
[local]Redback(config-atm-pvc)#bridge mac-entry 00:00:00:00:01:33
[local]Redback(config-atm-pvc)#bridge mac-entry 00:0a:0a:04:01:34
[local]Redback(config-atm-pvc)#bind interface if-isp1
```

The following selects an Ethernet port, configures it with an 802.1Q PVC, and configures the PVC as a tributary circuit for IPS3 subscribers:

```
[local]Redback(config)#port ethernet 2/1
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#mtu 1500
[local]Redback(config-port)#dot1q pvc 100
[local]Redback(config-dot1q-pvc)#bridge profile prof-sub-isp3-trib
[local]Redback(config-dot1q-pvc)#bridge mac-entry 00:00:00:00:01:31
[local]Redback(config-dot1q-pvc)#bridge mac-entry 00:0a:0a:04:01:32
[local]Redback(config-dot1q-pvc)#bind interface if-isp3
```

The following creates a named subscriber profile (`isp1`) and associates it with a bridge profile (`prof-sub-isp1-trib`) and a bridge (`isp1`):

```
[local]Redback(config)#context bridge
[local]Redback(config-ctx)#subscriber profile isp1
[local]Redback(config-sub)#bridge profile prof-sub-isp1-trib
[local]Redback(config-sub)#bridge name isp1
```

The following creates a subscriber record that has the named subscriber profile `isp1` associated with it. The named subscriber profile associates the subscriber with the bridge profile and the bridge:

```
[local]Redback(config)#context bridge
[local]Redback(config-ctx)#subscriber name 9991112222
[local]Redback(config-sub)#profile isp1
```

3.2 Filtered Bridged Ethernet Ports: Example

In the following example, two bridges are configured. One bridge, named `untag-bridge`, passes BPDU frames that are given the highest priority. In the second bridge, named `tagged-bridge`, the BPDU frames are dropped. The bridges segregate the BPDU traffic from the data traffic coming from the `vpls-context` context in which both bridges exist.

The configuration is divided into three sections to illustrate the basic steps in configuring bridged Ethernet ports.

1 In the first section, two bridge profiles are created with the needed BPDU properties. These profiles will be applied as needed to the bridged ports.

2 In the second section, a context is named and in it two bridges are created, each with a bridging-capable interface that will be bound the ports which are to be bridged.

3 The final section exits from the context and sets up two ports on the router so that each is bound to a bridging-capable interface and set with the bridging parameters provided the bridge profiles created earlier.

```
[local]Redback(config)#bridge profile untag
[local]Redback(config-bridge-profile)#bpdu allow-only
[local]Redback(config-bridge-profile)#bpdu priority 0
[local]Redback(config-bridge-profile)#exit
[local]Redback(config)#bridge profile tagged
[local]Redback(config-bridge-profile)#bpdu deny
[local]Redback(config-bridge-profile)#exit
!
[local]Redback(config)#context vpls-context
[local]Redback(config-ctx)#bridge untag-bridge
[local]Redback(config-bridge)#exit
[local]Redback(config-ctx)#interface untag-int bridge
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#bridge tagged-bridge
[local]Redback(config-ctx)#interface tagged-int bridge
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#exit
!
[local]Redback(config)#port ethernet 4/1
[local]Redback(config-port)#bridge profile untag
[local]Redback(config-port)#bind interface untag-int vpls-context
[local]Redback(config-port)#exit
[local]Redback(config)#port ethernet 4/2
[local]Redback(config-port)#bridge profile tagged
[local]Redback(config-port)#bind interface tagged-int vpls-context
```

3.3 Bridged VPLS Neighbors with RSTP Enabled: Example

In the two-router VPLS bridge in this example, the routers that are part of the bridge are connected by one or more pseudowires.

The following illustration shows provider edge routers PE1 and PE2 connected by a pseudowire (PW) on which the STP has been enabled. Section 3.3.1 on page 35 and Section 3.3.2 on page 37 show the configuration of PE1 and PE2. The configuration of the attachment circuits to customer edge routers CE1 and

CE2 is not provided, nor is the configuration of PE3 and PE4. Note that the attachment circuits to CE1 and CE2 are bound to interfaces on the VPLS bridge.

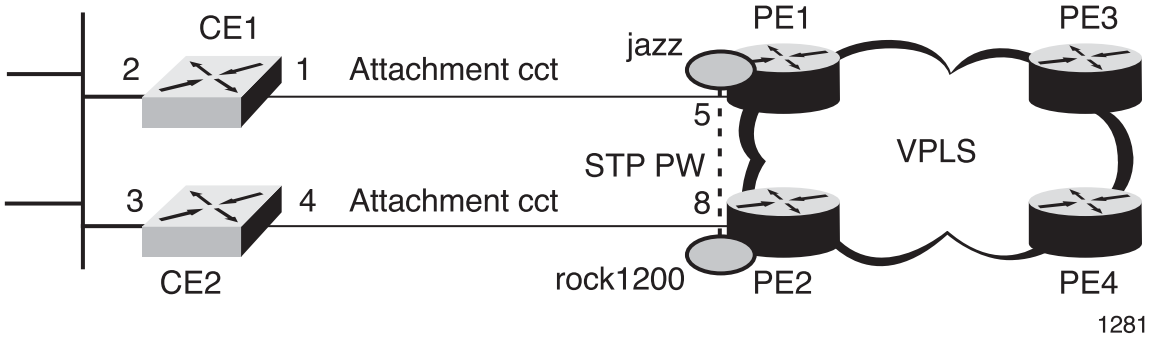

Figure 6 Bridged VPLS Neighbors with RSTP enabled

3.3.1 PE1 Router Configuration (rock1200)

The sections that configure STP, the bridge profile, the VPLS profile, the loopback interface, the interface `to-jazz`, MPLS routing, Label Distribution Protocol (LDP) routing, and the VPLS bridge are all essential to the VPLS bridge configuration. The other configuration attributes are incidental. The circuits or ports that carry traffic between VPLS neighbors are not shown in this configuration.

```
[local]Redback>show configuration
!
!
! STP global configuration
spanning-tree profile stp
 p2p-port
!
! Bridge global configuration
bridge profile trunk
 trunk
!
! VPLS global configuration
vpls profile foo
  neighbor 10.10.10.1
    bridge-profile trunk
    spanning-tree-profile  stp
!
context local
!
 interface loopback loopback
  ip address 10.10.10.2/32
!
 interface to-jazz
  ip address 1.1.1.2/24
 logging console
!
 router ospf 1
  fast-convergence
  area 0.0.0.0
   interface loopback
   interface to-jazz
!
 router mpls
  interface loopback
  interface to-jazz
!
 router ldp
  router-id 10.10.10.2
  neighbor 10.10.10.1 targeted
  interface loopback
  interface to-jazz
!
 bridge ted
  vpls
   pw-id 100
   profile foo
!
! ** End Context **
!
 logging tdm console
 logging active
 logging standby short
!
card ge-20-port 9
!
port ethernet 9/1
 no shutdown
 bind interface to-jazz local
!
no service console-break
service crash-dump-dram
no service auto-system-recovery
end
```

3.3.2 PE2 Router Configuration (jazz)

```
[local]Redback>show configuration
!
!
! STP global configuration
spanning-tree profile stp
 p2p-port
!
! Bridge global configuration
bridge profile trunk
 trunk
!
! VPLS global configuration
vpls profile foo
  neighbor 10.10.10.2
    bridge profile trunk
    spanning-tree-profile  stp
!
context local
!
 interface loopback loopback
  ip address 10.10.10.1/32
!
 interface to-rock1200
  ip address 1.1.1.1/24
!
 router ospf 1
  fast-convergence
  area 0.0.0.0
   interface loopback
   interface to-rock1200
!
 router mpls
  interface loopback
  interface to-rock1200
!
 router ldp
  router-id 10.10.10.1
  neighbor 10.10.10.2 targeted
  interface loopback
  interface to-rock1200
!
 bridge ted
!
  vpls
   pw-id 100
   profile foo
!
! ** End Context **
!
 logging tdm console
 logging active
 logging standby short
!
card ge-5-port 6
!
port ethernet 6/1
 no shutdown
 bind interface to-rock1200 local
!
port ethernet 7/1
! XCRP management ports on slot 7 and 8 are configured through 7/1
no shutdown
bind interface mgmt local
!
no service console-break
service crash-dump-dram
no service auto-system-recovery
end
```

3.3.3 VPLS Bridge Verification

To verify that the VPLS bridge peers are enabled and connected, enter the `show vpls peer` command. The state of is shown as down until packets are transmitted to it, which can be accomplished by pinging the VPLS neighbor.

```
[local]Redback#show vpls peer
 VPLS Bridge     Peer ID         Pseudo-wire ID  Circuit ID  Type   State
 ted             10.10.10.2      100             VPLS 1      Hub    Up
```

3.3.4 Show Status of Bridge Bindings

Use the `show bridge bindings all detail` command to show the status of the bridge bindings.

```
[local]Redback>show bridge bindings all detail
Flags are,(L)earning,(D)ynamic,(t)ributary,(T)runk,
(R)estricted,(d)eleted,(S)tale,(l)og,(U)p, (Ld) Loop Detection
(D)eny bpdu,(O)nly bpdu,(A)llow bpdu (default), (F) stp state forwarding
(B) stp state discarding
Headings : Ld Pri - Loop Detection Priority
          (Bpdu Pri)- bridge protocol data unit Priority
Context    Bridge Group Circuit     MAC   D-MAC Limit    Flag Ld Pri    Bpdu Pri
local      ted          VPLS 1      0     0     524290   L-T----U--AF   0    -

    Source Filter Name      : -
    Source Filter Id        : 0
```

3.3.5 Show Spanning Tree Status

Use the `show spanning-tree bridge` command to show the status of the spanning tree in the bridge at the port or circuit level. An example of this command used in a different RSTP configuration is shown in Section 3.4.2 on page 40.

3.4 Using RSTP Tracking for Cisco Interoperating: Example

Figure 7 illustrates a network in which the SmartEdge router is configured as a bridge that interoperates with a Cisco switch.

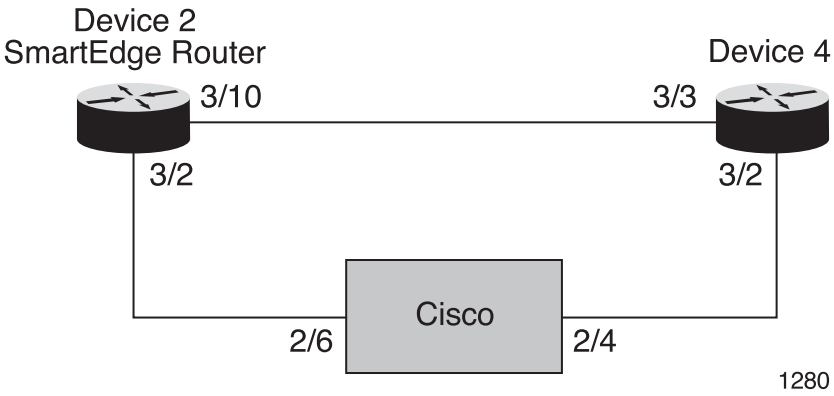

Figure 7 Cisco Interoperating Example

3.4.1 Cisco Interoperating Configuration

The following configuration applies to Device 2 in Figure 7. Because bridge `bridge2` tracks the RSTP master `bridge30`, the blocking, forwarding, or flushing action on the untagged interfaces `int_bridge30` ports 3/2 and 3/3 are tracked by `int_bridge2` on the corresponding ports.

```
Current configuration:
!
!
!
!
service multiple-contexts
!
!
! STP global configuration
spanning-tree profile sp1
 p2p-port
!
!
! Bridge global configuration
bridge profile bprofile1
 trunk
!
!
!
!
!
context local
!
 no ip domain-lookup
!
 interface int_bridge2 bridge
  bridge name bridge2
!
 interface int_bridge30 bridge
  bridge name bridge30
!
 interface mgmt
  ip address 10.12.48.49/23
 no logging console
!
 enable encrypted 1 $1$........$kvQfdsjs0ACFMeDHQ7n/o.
!
 aaa authentication administrator local !
 administrator test encrypted 1 $1$........$kvQfdsjs0ACFMeDHQ7n/o.
   timeout session idle 99999
!
!
 ip route 10.0.0.0/8 10.12.49.254
 ip route 155.53.0.0/16 10.12.49.254
```

```
!
 bridge bridge2
  track spanning-tree bridge30 local
!
!
 bridge bridge30
!
  spanning-tree
   master
   priority 0
!
!
! ** End Context **
 no logging tdm console
 no logging active
 no logging standby
!
!
!
card ge-10-port 3
!
port ethernet 3/2
 no shutdown
 encapsulation dot1q
 bind interface int_bridge30 local
  spanning-tree profile sp1
 dot1q pvc 2
  bind interface int_bridge2 local
  bridge profile bprofile1
!
port ethernet 3/3
 no shutdown
 encapsulation dot1q
 bind interface int_bridge30 local
  spanning-tree profile sp1
 dot1q pvc 2
  bind interface int_bridge2 local
  bridge profile bprofile1
!
!
port ethernet 7/1
! XCRP management ports on slot 7 and 8 are configured through 7/1
no shutdown
bind interface mgmt local !
card ge-10-port 9
!
card ge-10-port 13
!
!
 system hostname Redback
!
 boot configuration currentstpcisco
 no timeout session idle
!
no service console-break
!
service crash-dump-dram
!
no service auto-system-recovery
!
service upload-coredump ftp://arts:/4C4DF8713BA342DD/@155.53.12.41/se_io_coredump
!
end
```

3.4.2 Show Spanning-Tree Status

Use the `show spanning-tree bridge` command to show the status of the spanning tree in the bridge at the port or circuit level. The following example shows the status of Device 4 in Figure 7.

```
[local]Redback#show spanning-tree bridge
```

```
Context     Bridge      Bridge              STP   Bridge  Root                Admin
Name        Name        MAC                 Type  Pri     Bridge              State

local       bridge30    00:02:03:04:16:50   RSTP  4096    00:30:88:04:16:52   Enable
!
[local]Redback#show spanning-tree bridge30 circuit

Circuit                   Circuit           Port    Admin   Oper  STP
Identifier                Mac Address       Number  State   State State
3/2                       00:30:88:13:2a:a4 2       Enable  Up    FORWARD
3/3                       00:30:88:13:2a:a5 3       Enable  Up    FORWARD
!
[local]Redback#show spanning-tree bridge30 circuit detail

Circuit               : 3/2                   Cct State       : Up
Internal handle       : 3/2:1023:63/1/1/15
Bridge Name           : bridge30              Context Name    : local
:::STP Port Information:::
  STP Admin Status    : Enable                STP Oper Status : Up
  STP State           : FORWARD               RSTP Role       : DESIGNATED
  Port Number         : 2                     Port Priority   : 128
  Path Cost           : 4                     Root Path Cost  : 4
  P2P Port            : Yes                   Edge Port       : NO
  Oper Version        : RSTP                  Topo Chg Status : 0
:::Designated Bridge Info :::
  Root Bridge         : (MAC: 00:30:88:04:16:52 Prio:     0)
  Designated Bridge   : (MAC: 00:02:03:04:16:50 Prio:  4096)
  Designated Port     : (Number: 2 Priority: 128)
:::Circuit Information:::
  Source MAC          : 00:30:88:13:2a:a4     Group MAC       : 01:80:c2:00:00:00
  Encap Type          : ethernet
:::Statistics:::
  Last Topo Chg (secs): 3798                  Num Topo Changes: 1
  Last Tx (secs)      : 1
  BPDU Sent           : 0                     BPDU Rcvd       : 0
  TCN BPDU Sent       : 0                     TCN BPDU Rcvd   : 0
  RSTP BPDU Sent      : 1940                  RSTP BPDU Rcvd  : 3
  MST BPDU Sent       : 0                     MST BPDU Rcvd   : 0
  Expired BPDUs       : 0
:::State Transition Log:::
  State: DISABLE    Time: Feb 4 14:03:17.765.894
  State: DISCARD    Time: Feb 4 14:03:17.822.248
  State: FORWARD    Time: Feb 4 14:03:26.898.485
:::State Machine Timers:::
  Hello               : Expires in 1 secs 39 msecs
  Hold                : Expires in 0 secs 56 msecs


Circuit               : 3/3                   Cct State       : Up
Internal handle       : 3/3:1023:63/1/1/18
Bridge Name           : bridge30              Context Name    : local
:::STP Port Information:::
  STP Admin Status    : Enable                STP Oper Status : Up
  STP State           : FORWARD               RSTP Role       : ROOT
  Port Number         : 3                     Port Priority   : 128
  Path Cost           : 4                     Root Path Cost  : 4
  P2P Port            : Yes                   Edge Port       : NO
  Oper Version        : RSTP                  Topo Chg Status : 0
:::Designated Bridge Info :::
  Root Bridge         : (MAC: 00:30:88:04:16:52 Prio:     0)
  Designated Bridge   : (MAC: 00:30:88:04:16:52 Prio:     0)

Designated Port     : (Number: 3 Priority: 128)
:::Circuit Information:::
  Source MAC          : 00:30:88:13:2a:a5     Group MAC       : 01:80:c2:00:00:00
  Encap Type          : ethernet
:::Statistics:::
  Last Topo Chg (secs): 3794                  Num Topo Changes: 1
  Last Tx (secs)      : 3792
  BPDU Sent           : 0                     BPDU Rcvd       : 0
  TCN BPDU Sent       : 0                     TCN BPDU Rcvd   : 0
  RSTP BPDU Sent      : 9                     RSTP BPDU Rcvd  : 1935
  MST BPDU Sent       : 0                     MST BPDU Rcvd   : 0
  Expired BPDUs       : 0
:::State Transition Log:::
```

```
  State: DISABLE    Time: Feb 4 14:03:17.822.817
  State: DISCARD    Time: Feb 4 14:03:20.921.973
  State: FORWARD    Time: Feb 4 14:03:30.400.297
:::State Machine Timers:::
  Edge Delay          : Expires in 2 secs 22 msecs
  Hello               : Expires in 1 secs 39 msecs
  Message Age         : Expires in 4 secs 971 msecs
```

3.4.3 Show RSTP Tracking Master and Clients Status

Use the `show bridge bindings all tracking` command to show the tracking status of circuits bound to the bridge. The following example shows the tracking status for bridges configured in Device 4 in Figure 7.

```
[local]Redback#show bridge bindings all tracking

Context              Bridge Group         Circuit              Stp State    Role    clients
local                bridge30             3/2                  FORWARD      MASTER  1
local                bridge2              3/2 vlan-id 2        FORWARD      CLIENT  -
local                bridge30             3/3                  FORWARD      MASTER  1
local                bridge2              3/3 vlan-id 2        FORWARD      CLIENT  -
```

You can also use of the `show spanning-tree track` command to display tracking status:

```
[local]Redback#show spanning-tree track bridge30 local
Role   Bridge Group                  Context

Master bridge30                      local
clients:1    master cct count:2   master pw count:0   client cct count:2   client pw count:0

Client bridge2                       local
client cct count:2     client pw count:0
```

3.4.4 Cisco Switch Configuration

The following examples shows the configuration of ports 2/4 and 2/6 in the Cisco switch in Figure 7.

```
cat6k#show running-config interface GigabitEthernet2/4

Current configuration : 260 bytes
!
interface GigabitEthernet 2/4
 switchport
 switchport access vlan 2
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1,2
 switchport mode trunk
 logging event link-status
 flowcontrol receive on
 no cdp enable
 spanning-tree portfast
end

cat6k#show running-config interface GigabitEthernet2/6

Current configuration : 266 bytes
!
interface GigabitEthernet2/6
 switchport
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1,2
 switchport mode trunk
 logging event link-status
 flowcontrol receive on
 no cdp enable
 spanning-tree portfast
 spanning-tree port-priority 64
end


!
spanning-tree mst 0-1 priority 24576
spanning-tree vlan 1-2 priority 8192
```

3.5 BVI Port: Examples

BVI ports are created by inserting an IP interface with a virtual MAC into a bridge group to allow any bridging traffic sent to the BVI port to be routed and any IP traffic sent to the BVI port to be bridged. In typical BVI port implementations, a virtual IP interface is assigned to each bridge group that is used to route packets between the bridged and routed networks. This section explains how BVI ports can be used to support different types of networks. For example, you can configure BVI ports to terminate multiple types of Layer 2 circuits into a single bridge group, remove Ethernet switches to media gateways in a mobile-packet-based network, and eliminate physical loopback cables.

3.5.1 BVI Port: Basic Example

Figure 8 shows a BVI port connected to a standard Layer 2 bridging domain, called a bridge group in the operating system. In this case, a need exists to terminate multiple types of Layer 2 circuits into a single bridge group so that local or non-routable traffic is bridged among the bridged interfaces in the same bridge group, while traffic sent to the BVI port is routed.

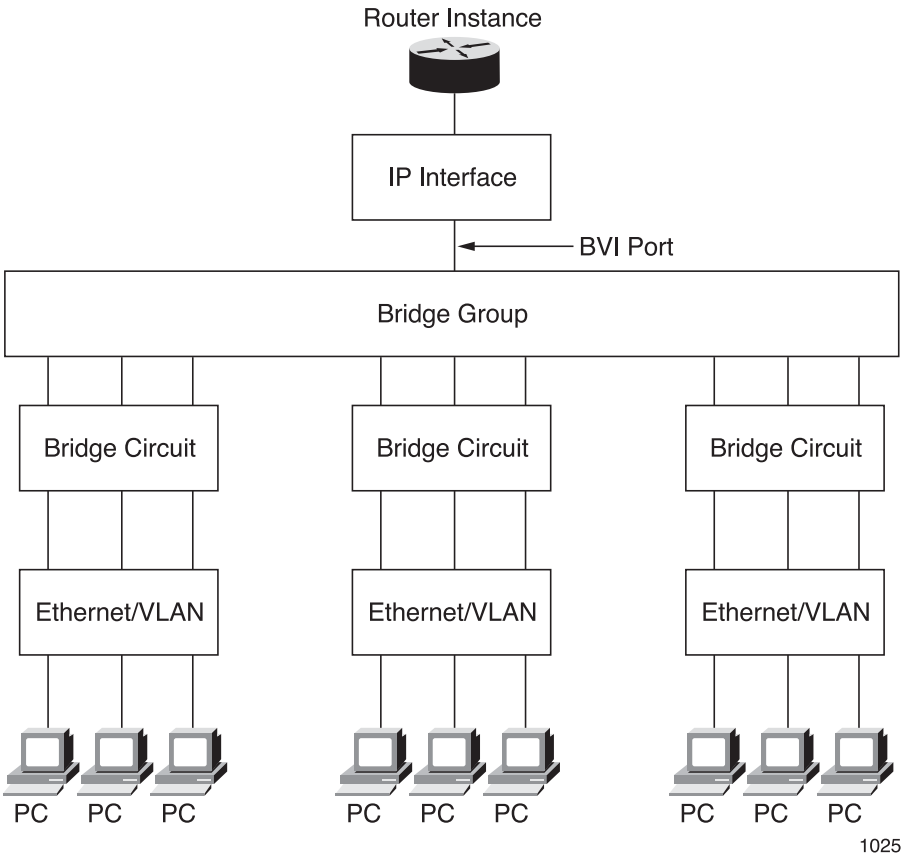

Figure 8 BVI Port Wireline Configuration

3.5.2 BVI Bridge with RSTP Enabled

In a ring topology where multiple bridges are linked to a BVI bridge, the BVI bridge serves as the default Layer 3 gateway. When a bridging loop is detected, RSTP breaks the loops by breaking the link between bridges in the RSTP group.

The following illustration and corresponding CLI shows an example of a ring topology and the configuration of the BVI bridge, which is a member of the RSTP group:

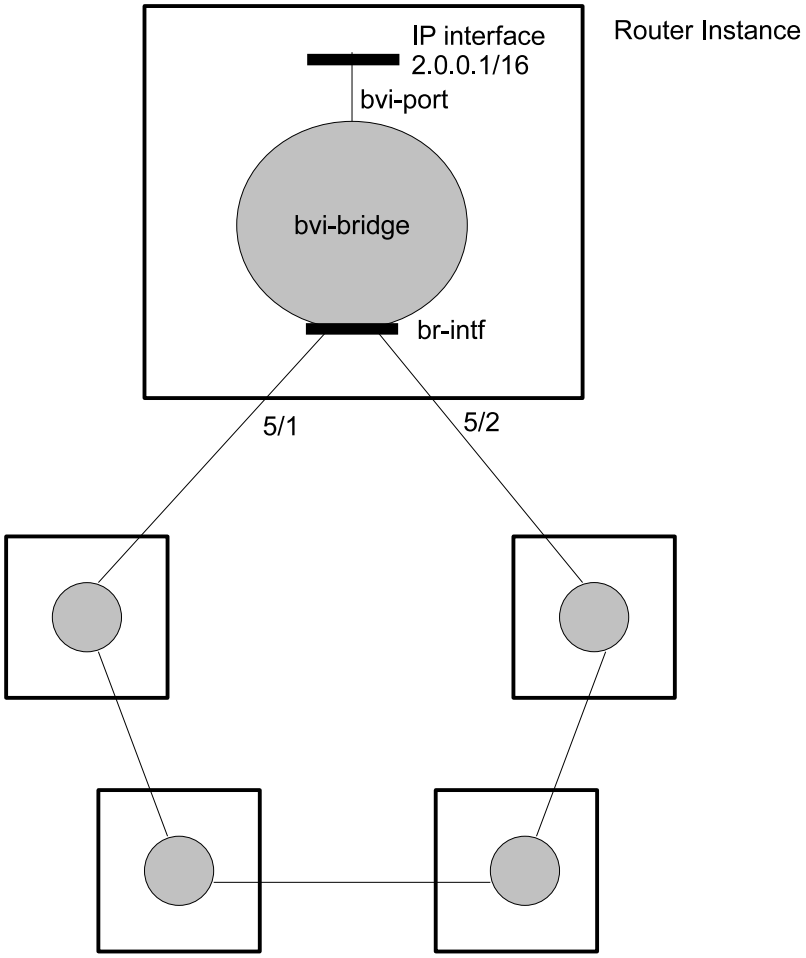

Figure 9 BVI Bridge in an RSTP Group

The CLI configuration for the router instance containing the `bvi-bridge` follows. The configuration of the bridging circuits and the configuration of the other members of the RSTP group is not provided:

```
config
 spanning-tree profile rstp
  p2p-port
 bridge profile trunk
  trunk

 context single-bvi                      // See Note 1
  interface br-intf bridge               // See Note 2
   bridge name bvi-bridge
  interface bvi-intf                     // See Note 3
   ip address 2.0.0.1/16
  interface core                         // See Note 3
   ip address 1.0.0.1/16
  bridge bvi-bridge                      // See Note 4
   bridge-mac-address 00:30:02:00:00:01
   spanning-tree
    priority 8192

 port ethernet 5/1
  no shutdown
  bind interface br-intf single-bvi      // See Note 5
   bridge profile trunk
   spanning-tree profile rstp

 port ethernet 5/2
  no shutdown
  bind interface br-intf single-bvi      // See Note 5
   bridge profile trunk
   spanning-tree profile rstp

 port ethernet 6/10
  no shutdown
  bind interface core single-bvi         // See Note 6

 port bvi bvi-port
  no shutdown
  bind interface bvi-intf single-bvi     // See Note 7
  bridge name bvi-bridge single-bvi
end
```

1 Context of BVI bridge

2 Bridged interface bound to BVI bridge

3 IP addresses assigned to bridged and nonbridged interfaces

4 BVI bridge with RSTP

5 Bridged interface bound to ports connecting to the network core

6 Nonbridged routing interface with IP address bound to a port facing CPEs

7 Nonbridged interface bound to the BVI port (network core-facing) bound to the BVI bridge

3.5.3 Bridged Virtual Interface Port: Example

This section includes examples that show how to create a simple bridged virtual interface port by using Ethernet interfaces and a bridged interface port by using a bridge over a non-economical access type link group.

The command lines show how to create a bridged virtual interface port, create a bridge group, create a bridge interface in a context, create an IP interface in the same or a different context, create a BVI port that is associated with the bridge group and the IP interface, and associate the bridge interface with two Ethernet ports.

1 Create a bridge group named `bvi-bridge` in a context named `bvi-context`.

2 Create a bridge interface named `br1` in a context named `bvi-context`.

3 Create an IP interface named `i1` in the same context.

4 Create a BVI port named `port-bvi` that is associated with the bridge group named `bvi-bridge` and the IP interface named `i1`.

5 Associate the bridge interface named `br1` with two Ethernet ports (`eth 2/2` and `eth 2/1`).

```
[local]Redback(config)#context bvi-context
[local]Redback(config-ctx)#bridge bvi-bridge
[local]Redback(config-bridge)#description Bridge for BVI to support routed and bridged traffic

[local]Redback(config-ctx)#interface i1
[local]Redback(config-if)#ip address 192.168.110.1 255.255.255.0

[local]Redback(config-ctx)#interface br1 bridge
[local]Redback(config-if)#bridge name bvi-bridge

[local]Redback(config)#port bvi port-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface i1 bvi-context
[local]Redback(config-port)#bridge name bvi-bridge bvi-context

[local]Redback(config)#port eth 1/1
[local]Redback(config-port)#bind interface br1 bvi-context

[local]Redback(config)#port eth 2/1
[local]Redback(config-port)#bind interface br1 bvi-context
```

The following command lines show how to create a bridged interface port by using a bridge over a non-economical access link group, create a bridge group, create a bridge interface in a context, create an IP interface in the same or a different context, create a non-economical access type link group, associate the bridge interface with the link group, create a BVI port that is associated with the bridge group and the IP interface, and create an Ethernet port and add it into the link group. (The slot mask of the circuits in this link group contain all the slots participating in the link group:)

1 Create a bridge group named `bvi-bridge2` in a context named `bvi-context`.

2 Create a bridge interface named `br2` in a context named `bvi-context`.

3 Create an IP interface named `ip` in the same context.

4 Create a non-economical access link group named `bvi-nealg`. Note that the access economical link group type is needed to configure economical access link groups, see *Configuring Link Aggregation* for more information.

5 Associate the bridge interface named `br2` with the `bvi-nealg` non-economical access link group.

6 Create a BVI port named `port-bvi` that is associated with the bridge group named `bvi-bridge2` and the IP interface named `ip`.

7 Create an 1/1 Ethernet port and add it into the link group named `bvi-nealg`.

The following commands illustrate this example:

```
[local]Redback(config)#context bvi-context
[local]Redback(config-ctx)#bridge bvi-bridge2
[local]Redback(config-bridge)#description Bridge over noneconomical
access link groups for BVI to support routed and bridged traffic

[local]Redback(config-ctx)#interface br2 bridge
[local]Redback(config-if)#bridge name bvi-bridge2

[local]Redback(config-ctx)#interface ip
[local]Redback(config-if)#ip address 10.1.1/24

[local]Redback(config)#link-group bvi-nealg access
[local]Redback(config)#qos hierarchical mode strict
[local]Redback(config)#encapsulation dot1q
[local]Redback(config)#dot1q pvc 1
[local]Redback(config)#bind interface br2 bvi-context

[local]Redback(config)#port bvi port-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface ip bvi-context
[local]Redback(config-port)#bridge name bvi-bridge2 bvi-context

[local]Redback(config)#port eth 1/1
[local]Redback(config-port)#no shut
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#link-group bvi-nealg
```

3.5.4 BVI Port: Media Gateway Connection Example

BVI ports can be used to bridge traffic to a Layer 2 network and route traffic into an IP/MPLS network, as shown in Figure 10. This eliminates the need for an Ethernet switch to the gateway.

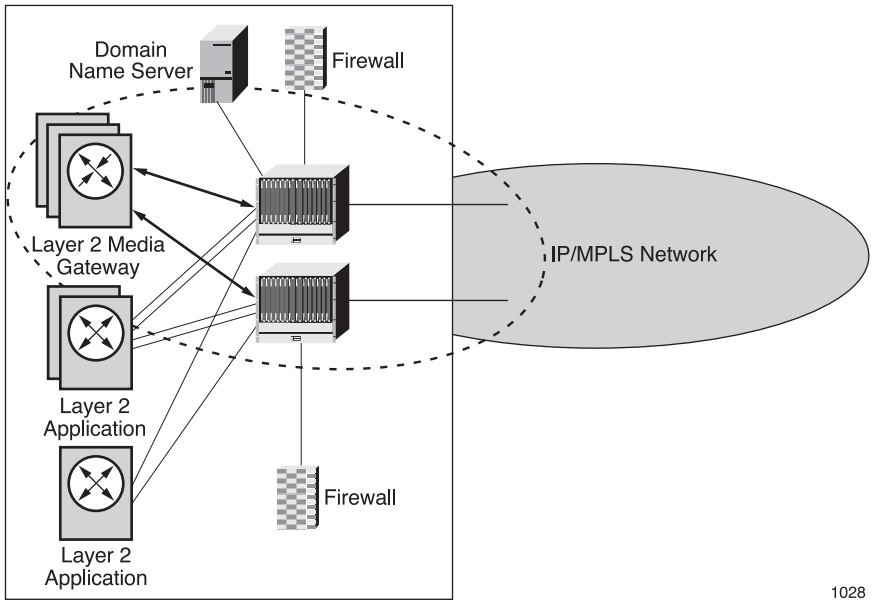

Figure 10 BVI Port Wireless Configuration

For example, the network configuration in Figure 10 can be created using BVI ports, instead of the configuration shown in Figure 11. For a sample configuration that shows a configuration before BVI port capability and after the BVI ports are configured, and the commands used to implement the BVI port configurations, see the following sections.

3.5.4.1 Pre-BVI Port Capability Sample

Figure 11 displays a sample media gateway connection configuration with high availability using the operating system on systems “SE800-1” and “SE800-2” before BVI port capability:

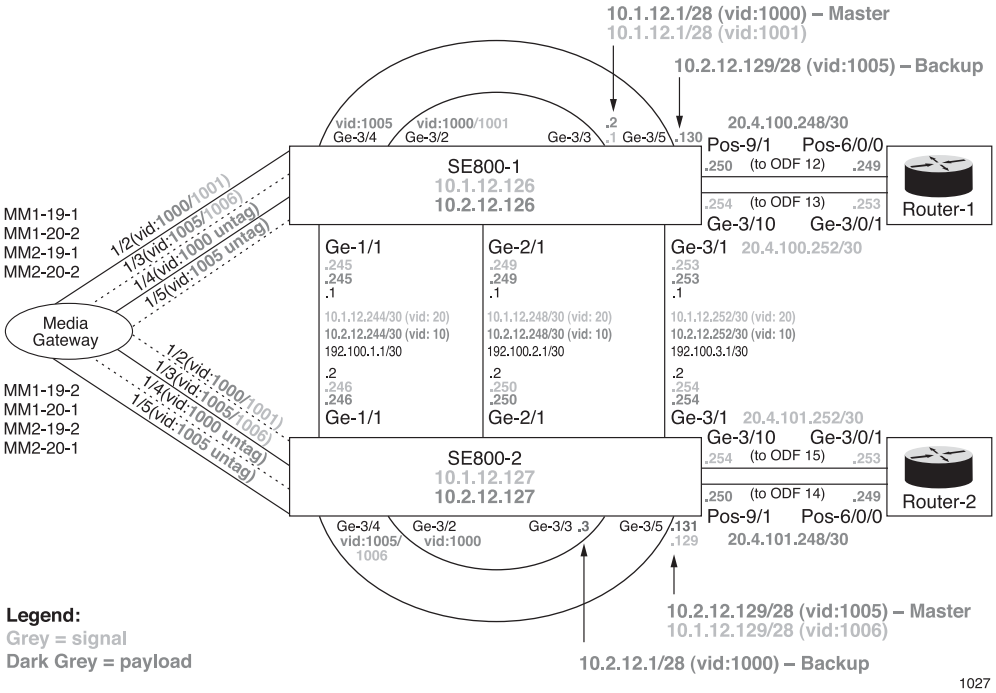

Figure 11 Sample Pre-BVI Port Capability Media Gateway Connection Configuration

The dotted lines to the media gateway in Figure 11 are the backup paths for each type of traffic (signal and payload) traveling through the gateway interface cards (MM1-20-2, MM2-20-2, MM1-19-2, MM2-19-2). The signal traffic supported is signal protocol over IP network applications that use Stream Control Transmission Protocol (SCTP). The payload traffic supported is Voice over IP (VoIP) network applications that use User Datagram Protocol (UDP).

The lines from SE800-1 Ge-1/1 to SE800-2 Ge-1/1, SE800-1 Ge-2/1 to SE800-2 Ge-2/1, and SE800-1 Ge-3/1 to SE800-2 Ge-3/1 are trunk links for VPLS over MPLS. These links allow L2 traffic and L3 traffic running between SE800-1 and SE800-2.

Notice the loopback cables on SE800-1 from Ge-3/4 to Ge-3/5 and from Ge-3/2 to Ge-3/3, and on SE800-2 from Ge-3/4 to Ge-3/5 and from Ge-3/2 to Ge-3/3. The physical loops are required in the pre-BVI port capability media connection configuration.

3.5.4.2 BVI Port Capability Sample

Figure 12 displays a sample media gateway connection with BVI port capability, followed by the commands used to configure the BVI ports, and how to complete the SE800-1 and SE800-2 configurations:

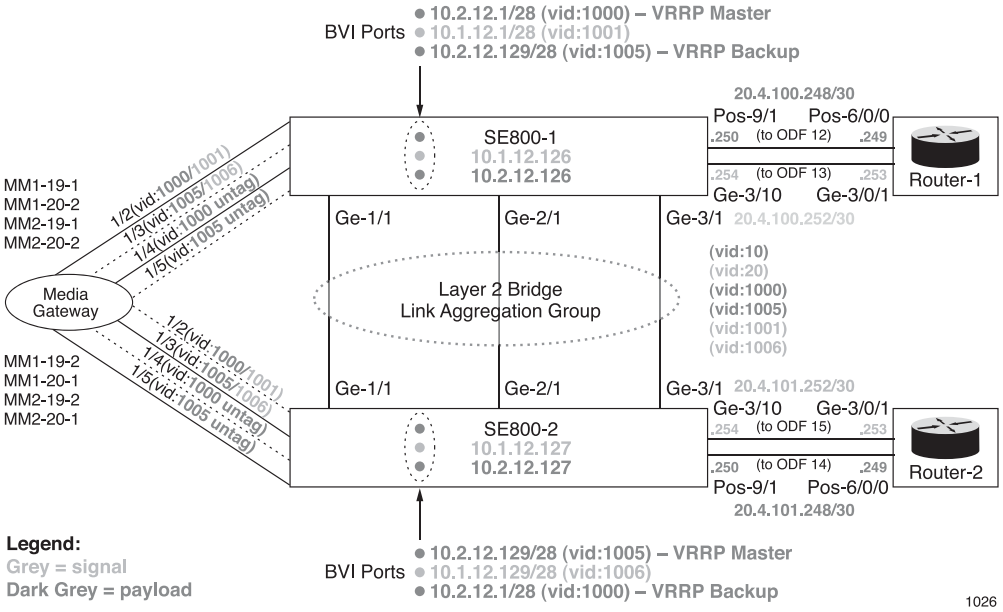

Figure 12 Sample BVI Port Capability Media Gateway Connection Configuration

Notice that the loopback cables on SE800-1 from Ge-3/4 to Ge-3/5 and from Ge-3/2 to Ge-3/3, and on SE800-2 from Ge-3/4 to Ge-3/5 and from Ge-3/2 to Ge-3/3 in the pre-BVI capability configuration shown in Figure 11 are no longer needed in the BVI port configuration in Figure 12.

The BVI port configuration in Figure 12 also eliminates the need for the Ethernet switches to the media gateway in Figure 11. Instead, if the traffic destination is the VMAC of the BVI, the traffic is routed; if the traffic destination is not the VMAC of the BVI, the traffic is bridged as usual on a standard bridge group.

The lines from SE800-1 Ge-1/1 to SE800-2 Ge-1/1, SE800-1 Ge-2/1 to SE800-2 Ge-2/1, and SE800-1 Ge-3/1 to SE800-2 Ge-3/1 that are the trunk links for VPLS over MPLS in Figure 11 are replaced with links for the Layer 2 Link Aggregation Group (L2 LAG) to allow L2 traffic and L3 traffic running between SE800-1 and SE800-2 in Figure 12.

The BVI ports support local or nonroutable Layer 2 (signal and payload) traffic that is bridged to the bridged interfaces in the same bridge group, while routable Layer 3 (signal and payload) traffic is routed to other routed interfaces. The SE800-1 system has three BVI ports: one for the signal tunnel VLAN associated with vid1001 and two for the payload tunnel VLANs, the active VLAN associated with vid1000, and the standby VLAN associated with vid1005. The SE800-2 system also has three BVI ports: one for the signal tunnel VLAN with vid1006 and two for the payload tunnel VLANs, the active VLAN associated with vid1005, and the standby VLAN associated with vid1000.

The following example shows how to implement the BVI port configuration in Figure 12 for SE800-1:

```
[local]Redback(config)#context xyz_Media
[local]Redback(config-ctx)#bridge bridge-vlan1000
```

```
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1000 routed and bridged traffic

[local]Redback(config-ctx)#bridge bridge-vlan1005
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1005 routed and bridged traffic

[local]Redback(config-ctx)#interface vlan1000
[local]Redback(config-if)#ip address 10.2.12.2/28

[local]Redback(config-ctx)#interface bridge-vlan1000 bridge
[local]Redback(config-if)#bridge name bridge-vlan1000

[local]Redback(config-ctx)#interface vlan1005
[local]Redback(config-if)#ip address 10.2.12.130/28

[local]Redback(config-ctx)#interface bridge-vlan1005 bridge
[local]Redback(config-if)#bridge name bridge-vlan1005

[local]Redback(config)#port bvi vlan1000-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1000 xyz_Media
[local]Redback(config-port)#bridge name bridge-vlan1000 xyz_Media

[local]Redback(config)#port bvi vlan1005-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1005 xyz_Media
[local]Redback(config-port)#bridge name bridge-vlan1005 xyz_Media


[local]Redback(config)#context xyz_SG

[local]Redback(config-ctx)#bridge bridge-vlan1001
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1001 routed and bridged traffic

[local]Redback(config-ctx)#bridge bridge-vlan1006
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1006 routed and bridged traffic

[local]Redback(config-ctx)#interface vlan1001
[local]Redback(config-if)#ip address 10.1.12.1/28

[local]Redback(config-ctx)#interface bridge-vlan1001 bridge
[local]Redback(config-if)#bridge name bridge-vlan1001

[local]Redback(config-ctx)#interface bridge-vlan1006 bridge
[local]Redback(config-if)#bridge name bridge-vlan1006

[local]Redback(config)#port bvi vlan1001-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1001 xyz_SG
[local]Redback(config-port)#bridge name bridge-vlan1001 xyz_SG


[local]Redback(config)#port eth 1/2
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 1000
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media
[local]Redback(config-port)#dot1q pvc 1001
[local]Redback(config-port)#bind interface bridge-vlan1001 xyz_SG

[local]Redback(config)#port eth 1/3
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 1005
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media
[local]Redback(config-port)#dot1q pvc 1006
[local]Redback(config-port)#bind interface bridge-vlan1006 xyz_SG

[local]Redback(config)#port eth 1/4
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media

[local]Redback(config)#port eth 1/5
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media
```

```
[local]Redback(config)#port eth 1/6
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media

[local]Redback(config)#port eth 1/7
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media
```

The following example shows the configuration for SE800-1 with the BVI ports configured in Figure 12

```
[local]Redback #show configuration
Building configuration...
!
!
!
service multiple-contexts
!
!
!
! Bridge global configuration
bridge profile bridge-trunk
trunk
!
!
context local
!
no ip domain-lookup
!
!
interface mgmt
ip address 192.100.200.1/30
!
!
administrator ericsson encrypted 1 $1$........$pkQOTEaa71lkHs4fdhZxyz
privilege start 15
!
service telnet
!
!
context xyz_Media   // Context for L2_Media_Gateway (voice payload)
!
no ip domain-lookup
!
interface loop loopback
ip address 10.2.12.126/32
!
interface to-AR1_Nb     // To AR1 (PE)
ip address 20.4.100.250/30
!
interface to-CE2_Nb     // to CE2 for L3 LAG
ip address 10.2.12.245/30
!
interface vlan1000     // VRRP group for Vlan 1000
ip address 10.2.12.2/28
vrrp 1 backup
virtual-address 10.2.12.1
advertise-interval millisecond 100
priority 100
preempt hold-time 120
!
interface vlan1005     // VRRP group for Vlan 1005
ip address 10.2.12.130/28
vrrp 2 backup
virtual-address 10.2.12.129
advertise-interval millisecond 100
priority 90
preempt hold-time 120
!
interface bridge-vlan1000 bridge     // Bridge port for Vlan 1000
bridge name bridge-vlan1000
!
interface bridge-vlan1005 bridge     // Bridge port for Vlan 1005
bridge name bridge-vlan1005
no logging console
```

```
!
router ospf 1     // OSPF between CEs and AR1
fast-convergence
router-id 10.2.12.126
graceful-restart
area 0.0.0.0
interface vlan1000    // VRRP vlan1000 interface advertised in ospf
cost 10
passive
interface vlan1005    // VRRP vlan1005 interface advertised in ospf
cost 10
passive
interface loop
interface to-CE2_Nb     // L3 LAG interface to CE2
network-type point-to-point
cost 10
interface to-AR1_Nb     // interface to AR1
network-type point-to-point
hello-interval 1
cost 10
authentication md5 CE-AR
redistribute static route-map CE1-to-AR1_Nb
!
ip prefix-list CE1-to-AR1_Nb
seq 10 permit 10.2.12.0/24 eq 24
!
route-map CE1-to-AR1_Nb permit 10
match ip address prefix-list CE1-to-AR1_Nb
!
key-chain CE-AR key-id 1
key-string encrypted C709B0E91B95E71666C3186385AF3XYZ
!
ip route 10.2.12.0/24 null0
service telnet  !
bridge bridge-vlan1000
bridge bridge-vlan1005
!
!
!
context xyz_SG         // Context for Signal
!
no ip domain-lookup
!
interface loop loopback
ip address 10.1.12.126/32
!
interface to-AR1_SG     // Connection to AR1 (PE)
ip address 20.4.100.254/30
!
interface to-CE2_SG     // to CE2 for L3 LAG
ip address 10.1.12.245/30
!
interface vlan1001     // Default Gateway for Vlan 1001
ip address 10.1.12.1/28
!
interface bridge-vlan1001 bridge     // Bridge port for vlan 1001
bridge name bridge-vlan1001
!
interface bridge-vlan1006 bridge     // Bridge port for vlan 1006
bridge name bridge-vlan1006
no logging console
!
router ospf 1     // OSPF between CEs and AR1
fast-convergence
router-id 10.1.12.126
graceful-restart
area 0.0.0.0
interface loop
interface vlan1001    // vlan1001 interface advertised in ospf
cost 10
passive
interface to-CE2_SG     // L3 LAG interface to CE2
network-type point-to-point
cost 10
interface to-AR1_SG     // interface to AR1
network-type point-to-point
```

```
hello-interval 1
cost 10
authentication md5 CE-AR
redistribute static route-map CE1-to-AR1_SG
!
ip prefix-list CE1-to-AR1_SG
seq 10 permit 10.1.12.0/25 eq 25
!
route-map CE1-to-AR1_SG permit 10
match ip address prefix-list CE1-to-AR1_SG
!
ip route 10.1.12.0/25 null0
service telnet
!
bridge bridge-vlan1001
bridge bridge-vlan1006
!
!
!
! ** End Context **
logging tdm console
logging active
logging standby
!
!
link-group CE1-CE2 dot1q     // L2 & L3 LAG between CEs
dot1q pvc 10     // L3 interface for signal
bind interface to-CE2_Nb xyz_Media
!
dot1q pvc 20     // L3 interface for payload
bind interface to-CE2_SG xyz_SG
!
dot1q pvc 1000     // L2 interface for payload
bind interface bridge-vlan1000 xyz_Media
!
dot1q pvc 1005     // L2 interface for payload
bind interface bridge-vlan1005 xyz_Media
!
dot1q pvc 1001     // L2 interface for signal
bind interface bridge-vlan1001 xyz_SG
!
dot1q pvc 1006     // L2 interface for signal
bind interface bridge-vlan1006 xyz_SG
!
!
card ge-10-port 1
!
port ethernet 1/1    // LAG to CE2
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
port ethernet 1/2    // To L2_Media_Gateway
no shutdown
encapsulation dot1q
dot1q pvc 1000
bind interface bridge-vlan1000 xyz_Media
dot1q pvc 1001
bind interface bridge-vlan1001 xyz_SG
!
port ethernet 1/3     // To L2_Media_Gateway
no shutdown
encapsulation dot1q
dot1q pvc 1005
bind interface bridge-vlan1005 xyz_Media
dot1q pvc 1006
bind interface bridge-vlan1006 xyz_SG
!
port ethernet 1/4     // To L2_Media_Gateway
no shutdown
bind interface bridge-vlan1000 xyz_Media
!
port ethernet 1/5     // To L2_Media_Gateway
no shutdown
bind interface bridge-vlan1005 xyz_Media
```

```
!
port ethernet 1/6
description Not connect L2_Media_Gateway - 20080422
no shutdown
bind interface bridge-vlan1000 xyz_Media
!
port ethernet 1/7
description Not connect L2_Media_Gateway - 20080422
no shutdown
bind interface bridge-vlan1005 xyz_Media
!
card ge-10-port 2
!
port ethernet 2/1     // LAG to CE2
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
card ge-10-port 3
!
port ethernet 3/1     // LAG to CE2
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
port ethernet 3/2
no shutdown
!
port ethernet 3/3
no shutdown
!
port ethernet 3/4
no shutdown
!
port ethernet 3/5
no shutdown
!
port ethernet 3/10      // To AR1 with signal
no auto-negotiate
link-dampening up 65535 down 0
no shutdown
bind interface to-AR1_SG xyz_SG
!
!
port ethernet 7/1
! XCRP management ports on slot 7 and 8 are configured through 7/1
 no shutdown
 bind interface mgmt local
!
card oc192-1-port 9
!
port pos 9/1     // To AR1 with payload

framing sdh
alarm-report-only ais-p rdi-p
no shutdown
encapsulation ppp
bind interface to-AR1_Nb xyz_Media
!
! BVI PORT global configuration
!
port bvi vlan1000-bvi     // BVI port for vlan 1000
no shutdown
bind interface vlan1000 xyz_Media
bridge name  bridge-vlan1000 xyz_Media
!
port bvi vlan1005-bvi     // BVI port for vlan 1005
no shutdown
bind interface vlan1005 xyz_Media
bridge name  bridge-vlan1005 xyz_Media
!
port bvi vlan1001-bvi     // BVI port for vlan 1001
no shutdown
bind interface vlan1001 xyz_SG
```

```
bridge name  bridge-vlan1001 xyz_SG
!
!
system hostname CQBB-PS-SX-CE01-SE800
no timeout session idle
!
no service console-break
!
service crash-dump-dram
!
no service auto-system-recovery
!
end
[local]Redback
```

The following example shows how to implement the BVI port configuration in Figure 12 for SE800-2:

```
[local]Redback(config)#context xyz_Media

[local]Redback(config-ctx)#bridge bridge-vlan1000
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1000 routed and bridged traffic

[local]Redback(config-ctx)#bridge bridge-vlan1005
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1005 routed and bridged traffic

[local]Redback(config-ctx)#interface vlan1000
[local]Redback(config-if)#ip address 10.2.12.3/28

[local]Redback(config-ctx)#interface bridge-vlan1000 bridge
[local]Redback(config-if)#bridge name bridge-vlan1000

[local]Redback(config-ctx)#interface vlan1005
[local]Redback(config-if)#ip address 10.2.12.131/28

[local]Redback(config-ctx)#interface bridge-vlan1005 bridge
[local]Redback(config-if)#bridge name bridge-vlan1005

[local]Redback(config)#port bvi vlan1000-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1000 xyz_Media
[local]Redback(config-port)#bridge name bridge-vlan1000 xyz_Media

[local]Redback(config)#port bvi vlan1005-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1005 xyz_Media
[local]Redback(config-port)#bridge name bridge-vlan1005 xyz_Media


[local]Redback(config)#context xyz_SG

[local]Redback(config-ctx)#bridge bridge-vlan1001
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1001 routed and bridged traffic

[local]Redback(config-ctx)#bridge bridge-vlan1006
[local]Redback(config-bridge)#description Bridge for BVI to support
vlan1006 routed and bridged traffic

[local]Redback(config-ctx)#interface vlan1006
[local]Redback(config-if)#ip address 10.1.12.129/28

[local]Redback(config-ctx)#interface bridge-vlan1001 bridge
[local]Redback(config-if)#bridge name bridge-vlan1001

[local]Redback(config-ctx)#interface bridge-vlan1006 bridge
[local]Redback(config-if)#bridge name bridge-vlan1006

[local]Redback(config)#port bvi vlan1006-bvi
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#bind interface vlan1006 xyz_SG
[local]Redback(config-port)#bridge name bridge-vlan1006 xyz_SG
```

```
[local]Redback(config)#port eth 1/2
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 1000
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media
[local]Redback(config-port)#dot1q pvc 1001
[local]Redback(config-port)#bind interface bridge-vlan1001 xyz_SG

[local]Redback(config)#port eth 1/3
[local]Redback(config-port)#no shutdown
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 1005
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media
[local]Redback(config-port)#dot1q pvc 1006
[local]Redback(config-port)#bind interface bridge-vlan1006 xyz_SG

[local]Redback(config)#port eth 1/4
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media

[local]Redback(config)#port eth 1/5
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media

[local]Redback(config)#port eth 1/6
[local]Redback(config-port)#bind interface bridge-vlan1000 xyz_Media

[local]Redback(config)#port eth 1/7
[local]Redback(config-port)#bind interface bridge-vlan1005 xyz_Media
```

The following example shows the configuration for SE800-2 with the BVI ports configured in Figure 12

```
[local]Redback #show configuration

Building configuration...
!
!
!
service multiple-contexts
!
!
!
! Bridge global configuration
bridge profile bridge-trunk
trunk
!
!
!
!
context local
!
no ip domain-lookup
!
!
interface mgmt
ip address 192.100.200.1/30
!
!
administrator ericsson encrypted 1 $1$........$pkQOTEaa71lkHs4fdhZxyz
privilege start 15
!
service telnet

!
!
context xyz_Media
!
no ip domain-lookup
!
interface loop loopback
ip address 10.2.12.127/32
!
interface to-AR2_Nb
```

```
ip address 20.4.101.250/30
!
interface to-CE1_Nb
ip address 10.2.12.246/30
!
interface vlan1000
ip address 10.2.12.3/28
vrrp 1 backup
virtual-address 10.2.12.1
advertise-interval millisecond 100
priority 90
preempt hold-time 120
!
interface vlan1005
ip address 10.2.12.131/28
vrrp 2 backup
virtual-address 10.2.12.129
advertise-interval millisecond 100
priority 100
preempt hold-time 120
!
interface bridge-vlan1000 bridge
bridge name bridge-vlan1000
!
interface bridge-vlan1005 bridge
bridge name bridge-vlan1005
no logging console
!
router ospf 1
fast-convergence
router-id 10.2.12.127
area 0.0.0.0
interface vlan1000
cost 10
passive
interface vlan1005
cost 10
passive
interface loop
interface to-CE1_Nb
network-type point-to-point
cost 10
interface to-AR2_Nb
network-type point-to-point
hello-interval 1
router-dead-interval 4
cost 10
authentication md5 CE-AR
redistribute static metric-type 1 route-map CE2-to-AR2_Nb
!
ip prefix-list CE2-to-AR2_Nb
seq 10 permit 10.2.12.0/24 eq 24
!
route-map CE2-to-AR2_Nb permit 10
match ip address prefix-list CE2-to-AR2_Nb
!
key-chain CE-AR key-id 1
key-string encrypted C709B0E91B95E71666C3186385AF3XYZ
!
ip route 10.2.12.0/24 null0
service telnet
!
bridge bridge-vlan1000
bridge bridge-vlan1005
!
!
!
context xyz_SG
!
no ip domain-lookup
!
interface loop loopback
ip address 10.1.12.127/32
!
interface to-AR2_SG
ip address 20.4.101.254/30
```

```
!
interface to-CE1_SG
ip address 10.1.12.246/30
!
interface vlan1006
ip address 10.1.12.129/28
!
interface bridge-vlan1001 bridge
bridge name bridge-vlan1001
!
interface bridge-vlan1006 bridge
bridge name bridge-vlan1006
no logging console
!
router ospf 1
fast-convergence
router-id 10.1.12.127
graceful-restart
area 0.0.0.0
interface loop
interface vlan1006
cost 10
passive
interface to-CE1_SG
network-type point-to-point
cost 10
interface to-AR2_SG
network-type point-to-point
hello-interval 1
router-dead-interval 4
cost 10
authentication md5 CE-AR
redistribute static metric-type 1 route-map CE2-to-AR2_SG
!
ip prefix-list CE2-to-AR2_SG
seq 10 permit 10.1.12.128/25 eq 25
!
route-map CE2-to-AR2_SG permit 10
match ip address prefix-list CE2-to-AR2_SG
!
key-chain CE-AR key-id 1
key-string encrypted C709B0E91B95E71666C3186385AF3XYZ
!
ip route 10.1.12.128/25 null0
service telnet
!
bridge bridge-vlan1001
bridge bridge-vlan1006
!
!
!
! ** End Context **
logging tdm console
logging active
logging standby short
!
!
link-group CE1-CE2 dot1q
dot1q pvc 10
bind interface to-CE1_Nb xyz_Media
!
dot1q pvc 20
bind interface to-CE1_SG xyz_SG
!
dot1q pvc 1000
bind interface bridge-vlan1000 xyz_Media
!
dot1q pvc 1005
bind interface bridge-vlan1005 xyz_Media
!
dot1q pvc 1001
bind interface bridge-vlan1001 xyz_SG
!
dot1q pvc 1006
bind interface bridge-vlan1006 xyz_SG
!
```

```
!
card ge-10-port 1
!
port ethernet 1/1
description to-CE1-ge1/1
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
port ethernet 1/2
description to-GM101-MM1-19-2
no shutdown


encapsulation dot1q
dot1q pvc 1000
bind interface bridge-vlan1000 xyz_Media
dot1q pvc 1001
bind interface bridge-vlan1001 xyz_SG
!
port ethernet 1/3
description to-GM101-MM1-20-1
no shutdown
encapsulation dot1q
dot1q pvc 1005
bind interface bridge-vlan1005 xyz_Media
dot1q pvc 1006
bind interface bridge-vlan1006 xyz_SG
!
port ethernet 1/4
description to-GM101-MM2-19-2
no shutdown
bind interface bridge-vlan1000 xyz_Media
!
port ethernet 1/5
description to-GM101-MM2-20-1
no shutdown
bind interface bridge-vlan1005 xyz_Media
!
port ethernet 1/6
description Not connect L2_Media_Gateway - 20080422
no shutdown
bind interface bridge-vlan1000 xyz_Media
!
port ethernet 1/7
description Not connect L2_Media_Gateway - 20080422
no shutdown
bind interface bridge-vlan1005 xyz_Media
!
card ge-10-port 2
!
port ethernet 2/1
description to-CE1-ge2/1
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
card ge-10-port 3
!
port ethernet 3/1
description to-CE1-ge3/1
no shutdown
encapsulation dot1q
bridge profile bridge-trunk
link-group CE1-CE2
!
port ethernet 3/2
description to-port 3/3
no shutdown
!
port ethernet 3/3
description to-port 3/2
no shutdown
!
port ethernet 3/4
```

```
description to-port 3/5
no shutdown
!
port ethernet 3/5
description to-port 3/4
no shutdown
!
port ethernet 3/10
description to-AR2_SG
no auto-negotiate
link-dampening up 65535 down 0
no shutdown
bind interface to-AR2_SG xyz_SG
!
!
port ethernet 7/1
! XCRP management ports on slot 7 and 8 are configured through 7/1
no shutdown
bind interface mgmt local
!
card oc192-1-port 9
!
port pos 9/1
description to-AR2_Nb
link-dampening up 65535 down 0
framing sdh
alarm-report-only ais-p rdi-p
no shutdown
encapsulation ppp
bind interface to-AR2_Nb xyz_Media
!
! BVI PORT global configuration
!
port bvi vlan1000-bvi
no shutdown
bind interface vlan1000 xyz_Media
bridge name  bridge-vlan1000 xyz_Media
!
port bvi vlan1005-bvi

no shutdown
bind interface vlan1005 xyz_Media
bridge name  bridge-vlan1005 xyz_Media
!
port bvi vlan1006-bvi
no shutdown
bind interface vlan1006
bridge name  bridge-vlan1006 xyz_SG
!
!
system hostname CQBB-PS-SX-CE02-SE800
no timeout session idle
!
no service console-break
!
service crash-dump-dram
!
no service auto-system-recovery
!
end
[local]Redback #
```

3.6 Validate IP Host Connectivity: Example

The following example shows how to initiate a CPE ping to discover an IP host address and troubleshoot the data path to the host. In this example, the CPE ping is successful, and no duplicate IP addresses are configured in the system:

```
[local]Redback# ping cpe 10.1.1.1 10.1.1.2 bridge br2 context local
PING 10.1.1.1 (10.1.1.1): source 10.1.1.2, 36 data bytes, timeout is 1
second !!!!!
----10.1.1.1 PING Statistics----
5 packets transmitted, 5 packets received, 0.0% packet loss round-trip
min/avg/max/stddev = 0.870/1.569/2.237/0.636 ms
```

The following example shows what happens when a CPE ping is terminated because duplicate IP addresses are detected:

```
[local]Redback#ping cpe 1.0.0.10 1.0.0.8 bridge br1


PING 1.0.0.10 (1.0.0.10): source 1.0.0.8
Duplicate IP Detected.
```

The following example shows what happens when an ARP ping is terminated because the PE does not receive an ARP response from the CPE:

```
[local]Redback#ping arp 10.1.1.3 bridge br2 source 10.1.1.2
source 10.1.1.2 5 Arp Request(s) sent to destination ip 10.1.1.3  with
source ip 10.1.1.2 but no reply
```

3.7 Bridge Profile: Example

The following example shows how to create a bridge profile for a restricted trunk (network-facing) circuit:

```
[local]Redback(config)#bridge profile prof-isp-trunk
[local]Redback(config-bridge-profile)#mac-limit 10
[local]Redback(config-bridge-profile)#restricted
[local]Redback(config-bridge-profile)#trunk
```

The following example shows how to create a bridge profile for an unrestricted tributary (access-facing) circuit:

```
[local]Redback(config)#bridge profile prof-sub-isp-trib
[local]Redback(config-bridge-profile)#mac-limit 10
[local]Redback(config-bridge-profile)#no trunk
```

3.8 Spanning Tree Profile: Example

The following example illustrates how the **spanning-tree profile** command creates the womp spanning-tree profile and sets its cost to 5000. In the second part of the example, an Ethernet port is assigned the womp spanning-tree profile and, therefore, the spanning-tree cost of bridging to the port is set at 5000:

```
[local]Redback(config)#spanning-tree profile womp
[local]Redback(config-stp-prof)#cost 5000
[local]Redback(config-stp-prof)#exit
[local]Redback(config)#port ethernet 1/1
[local]Redback(config-port)#spanning-tree profile womp
```

In the following example, the spanning-tree profile `womp` is assigned to a single 802.1Q permanent virtual circuit (PVC):

```
[local]Redback(config)#port ethernet 5/1
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 100
[local]Redback(config-dot1q-pvc)#spanning-tree profile womp
```

3.9 Bridged Profile for Trunk Circuits: Example

The following example shows how to select a Gigabit Ethernet port and configure it as a trunk circuit to `ISP1`:

```
[local]Redback(config)#port ethernet 5/1
[local]Redback(config-port)#bridge profile prof-isp-trunk
[local]Redback(config-port)#spanning-tree profile womp
[local]Redback(config-port)#mtu 1500
[local]Redback(config-port)#bridge mac-entry 00:d0:ba:04:d8:05
[local]Redback(config-port)#bridge mac-entry 00:0a:0a:04:d8:06
[local]Redback(config-port)#bind interface if-isp1
```

The following example shows how to select a Gigabit Ethernet port and configure it as a trunk circuit to `ISP3`:

```
[local]Redback(config)#port ethernet 5/3
[local]Redback(config-port)#bridge profile prof-isp-trunk
[local]Redback(config-port)#mtu 1500
[local]Redback(config-port)#bridge mac-entry 00:d0:ba:04:d8:07
[local]Redback(config-port)#bridge mac-entry 00:0a:0a:04:d8:08
[local]Redback(config-port)#bind interface if-isp3
```