ERICSSON

Configuring Cross-Connections

SYSTEM ADMINISTRATOR GUIDE

8/1543-CRA 119 1170/1-V1 Uen F

Contents

1 Overview

This document describes how to configure, monitor, and administer cross-connections between circuits with interfaces on the SmartEdge router.

Cross-connected circuits allow you to connect two circuits of the same or different type to pass traffic transparently through the SmartEdge router.

Note: This document does not cover cross connections over the network cloud. For the topics of *L2VPN cross connections* and *VPLS cross connections*, see the documents *Configuring L2VPN* and *Configuring VPLS*.

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 Terminology

The following terms are used in this document:

- In cross connections, one circuit is designated as the inbound circuit; the other is designated as the outbound circuit. An *inbound circuit* is on the upstream side of the SmartEdge router, while an *outbound circuit* is on the downstream side. In a provider-subscriber scenario, the subscriber is downstream from the provider; therefore, the *downstream* side of a cross connection faces the subscriber and the *upstream* side faces the provider.

- An *802.1Q PVC* is also referred to as an *802.1Q virtual LAN (VLAN)*; however, in this document, it is the PVC, not the VLAN, that is being cross-connected.

- Unless otherwise noted, the SmartEdge 100 router supports all commands described in this document.

- A *multiprotocol-encapsulated circuit* can carry child circuits with any combination of Point-to-Point Protocol over Ethernet (PPPoE) traffic, Internet Protocol over Ethernet (IPoE), or IP Version 6 (IPv6) over Ethernet

(IPv6oE) traffic; you can cross-connect the parent circuit and the child circuits.

- When IP Version 6 (IPv6) addresses are not referenced or explicitly specified, the term IP address can refer generally to IP Version 4 (IPv4) addresses, IPv6 addresses, or IP addressing. In instances where IPv6 addresses are referenced or explicitly specified, the term *IP address* refers only to IPv4 addresses. For a description of IPv6 addressing and the types of IPv6 addresses, see RFC 3513, *Internet Protocol Version 6 (IPv6) Addressing Architecture*.

1.2 Types of Non-Interworking Cross-Connections

The following sections describe the various types of non-interworking cross-connections. Only 802.1Q and Asynchronous Transfer Mode (ATM) permanent virtual circuits (PVCs), including those that you have configured with multiprotocol encapsulation, can be cross-connected.

1.2.1 ATM PVCs to ATM PVCs

When cross-connecting an ATM PVC to another ATM PVC, both PVCs must have the same type of encapsulation. Encapsulation types include RFC 1483 bridged, RFC 1483 routed, multiprotocol (for ATM parent circuits), PPPoE, or raw mode. Traffic is passed through the SmartEdge router without filtering because the encapsulations always match.

1.2.2 ATM PVCs to 802.1Q PVCs

The following combinations of ATM PVCs and 802.1Q PVCs, with and without child circuits, can be cross-connected, and filtering can occur:

- An ATM PVC with **`bridge1483`** encapsulation to an 802.1Q PVC with **`dot1q`** or **`raw`** encapsulation.
- An ATM PVC with **`multi`** encapsulation to an 802.1Q PVC with **`dot1q`** or **`multi`** encapsulation.
- An ATM PVC with **`pppoe`** encapsulation to an 802.1Q PVC with **`dot1q`** or **`pppoe`** encapsulation, or to the **`pppoe`**-encapsulated child circuit of an 802.1Q PVC with **`multi`** encapsulation.
- An ATM PVC child circuit with **`pppoe`** encapsulation or **`ipv6oe`** encapsulation to an 802.1Q PVC with **`dot1q`** or **`pppoe`** encapsulation, or to the **`pppoe`**-encapsulated or **`ipv6oe`**-encapsulated child circuit of an 802.1Q PVC with **`multi`** encapsulation.

ATM PVCs can also be cross-connected to 802.1Q tunnels and to 802.1Q PVCs within those tunnels and to 802.1Q tunnels and PVCs that are configured

within access link groups. For more information about access link groups and how to configure them, see *Configuring Link Aggregation*.

1.2.3 802.1Q PVCs to 802.1Q PVCs

All encapsulations, including **`dot1q`**, **`multi`**, **`ipv6oe`**, and **`raw`** encapsulations, are supported.

Encapsulation for **`pppoe`** is also supported.

1.3 Filtering Traffic Using Non-Interworking Cross-Connections

Filtering traffic using non-interworking cross-connections is described in the following sections.

1.3.1 Filtering Using Child Circuits

A multiprotocol-encapsulated circuit (also referred to as the parent circuit) can carry any combination of Point-to-Point Protocol over Ethernet (PPPoE) traffic, Internet Protocol over Ethernet (IPoE), or IP version 6 over Ethernet (IPv6oE) traffic.

Note: The traffic for a parent circuit is IPoE, but the encapsulation keyword is **`multi`**.

Filtering of the traffic of a particular type on a parent circuit is provided for by creating a child circuit with that type of encapsulation. Thus, a multiprotocol ATM or 802.1Q PVC can be a parent to two child circuits in addition to the parent circuit: one for PPPoE traffic (**`pppoe`** encapsulation) and one for IPv6oE traffic (**`ipv6oe`** encapsulation). In addition, the parent circuit carries IPoE traffic (**`ipoe`** encapsulation).

You can cross-connect a child circuit to another child circuit or to a circuit that has no child circuits.

When an inbound circuit is cross-connected to an outbound circuit with the same type of encapsulation, the inbound traffic is filtered, with only the packets with the designated encapsulation being passed to the outbound circuit.

To support child circuits, an ATM or 802.1Q PVC must be configured with multiprotocol encapsulation, as described *Configuring Circuits*.

Figure 1 shows the packet flow across a multiprotocol ATM PVC and how the traffic is split between IPoE, IPv6oE, and PPPoE destinations. In this example, the inbound IPoE parent circuit is terminated and routed to a Gigabit Ethernet port.

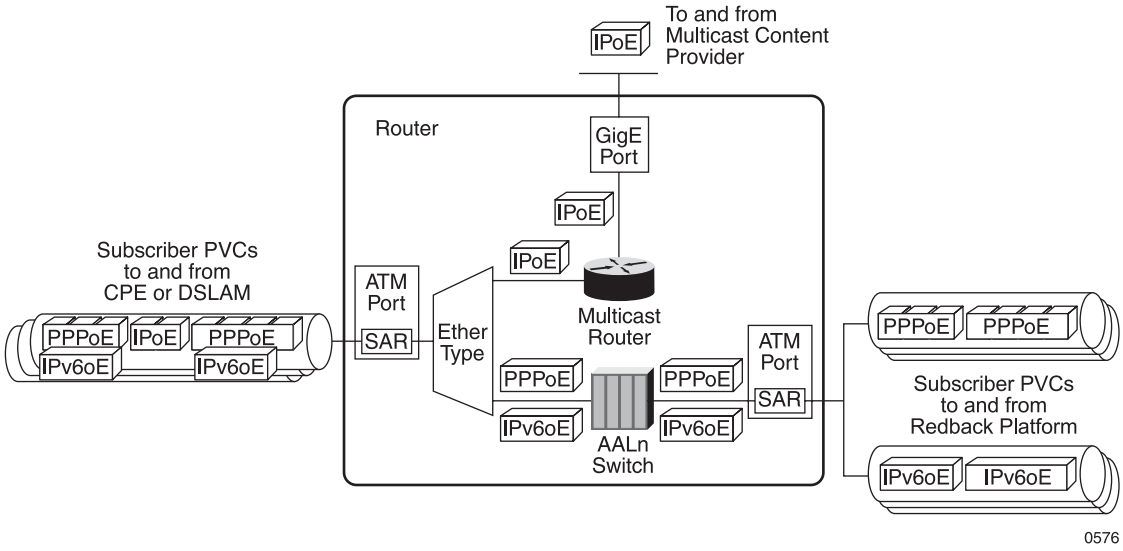

Figure 1 Traffic Paths for a Multiprotocol ATM PVC

1.3.2 Filtering Using Circuits with Unlike Encapsulations

Usually, you configure a cross-connection to pass through traffic of a particular type. In this case, the encapsulation of both the inbound and outbound circuits is the same and no other filtering takes place.

For example, an inbound child circuit with PPPoE traffic on an ATM PVC passes only PPPoE traffic to an outbound circuit. If the outbound circuit is a child circuit with PPPoE encapsulation, all the inbound PPPoE packets flow across the connection in both directions.

However, when the encapsulations of the circuits that form the cross-connection are different, the cross-connection acts as a filter on the traffic passing from the inbound circuit to the outbound circuit, and conversely.

For example, for a cross-connection between an inbound child circuit with PPPoE encapsulation and an outbound 802.1Q PVC with dot1q encapsulation, the following filtering actions occur:

- In the inbound-to-outbound direction:
  - Because the inbound child circuit has only PPPoE traffic, all other traffic on the parent circuit is ignored.
  - Because the **`dot1q`** encapsulation accepts PPPoE traffic, no packets are discarded; all PPPoE packets are transmitted over the 802.1Q PVC.
- In the outbound-to-inbound direction:
  - Because the **`dot1q`** encapsulation accepts all IP traffic, no packets are discarded.

– Because the child circuit accepts only PPPoE traffic, any non PPPoE packets are discarded.

1.4 Interworking Cross-Connections

Interworking cross-connections allow you to receive IPv4 frames from an ATM PVC with RFC 1483 routed encapsulation and transmit them on an 802.1Q PVC, including an 802.1Q PVC within 802.1Q tunnel.

Figure 2 shows an interworking network. The IPv4 device at each end is configured with ATM PVCs or 802.1Q PVCs. The SmartEdge router provides the interworking cross-connection between the two types of PVCs.

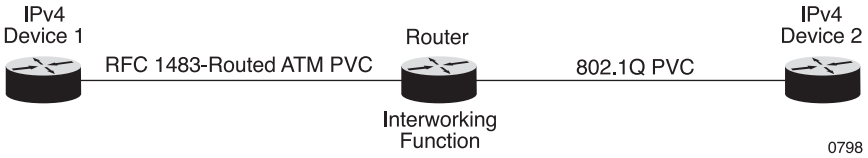

Figure 2 Interworking Network

Because the IPv4 device on the ATM PVC side is not capable of Address Resolution Protocol (ARP) functions, the SmartEdge router provides the following functions for the IPv4 device on the 802.1Q PVC side:

- Responds to its ARP queries
- Generates periodic ARP requests to learn about its medium access control (MAC) address
- Generates gratuitous ARP packets if any MAC changes occur

All non-ARP packets are simply cross-connected.

Configuration tasks for this type of cross-connection are described in the section “Cross-Connecting Circuits.”

1.5 Cross-Connect Groups

A cross-connect group is a named collection of cross-connections; it is created by using the **`xc-group`** command (in global configuration mode). The group is populated when a cross-connection is created through the **`xc`** command (in XC group configuration mode). The system maintains a default XC group, which you can specify instead of a named group when creating a cross-connection. You can add cross-connections to an existing group and remove cross-connections from it. You can also delete XC groups. You can add cross-connections of any type to any XC group.

1.6 Related Information

To create a cross connection for a L2VPN, see *Configuring L2VPN*.

Configuration for ATM and 802.1Q PVCs is described in *Configuring Circuits*.

2 Configuration and Operations Tasks

2.1 Configuration Guidelines for Cross-Connecting Circuits

Review the following guidelines for port and circuit configuration before you configure the cross-connection.

- *Supported Cross-Connections and Their Encapsulations* - (under the *xc* command description) lists the supported cross connections.
- To route traffic on child circuits that are not cross-connected, you must configure ports and circuits capable of supporting that type of traffic and bind each port and circuit to an interface.

  For child circuits, ports and circuits can be of any type that support IP packets; for example, Figure 1 shows a Gigabit Ethernet port that is configured for this type of traffic.
- You can configure a PVC with multiprotocol encapsulation and cross-connect it without creating child circuits on it. A PVC with multiprotocol encapsulation carries IPoE traffic.
- Ensure that you have bound each circuit to be cross-connected with the *bind bypass* command (in the configuration mode for that circuit).
- Circuits in any named XC group (or the default XC group) must be both bound (*bind bypass* command) and connected (*xc* command) in order for traffic to pass between them.
- You can specify either circuit being cross-connected as the inbound circuit; the cross-connection is symmetrical.

2.2 Configuration of Cross-Connecting Circuits

To cross-connect two circuits, perform the tasks in Table 1.

Table 1 Cross-Connect Two PVCs

Step	Task	Root Command	Notes
1.	Bind the circuit to the cross connection.	*bind bypass*	If you do not enter this command when configuring the PVC or child circuit, the configuration is incomplete and the status of the cross-connection is not displayed in the output of the **`show bypass`** command. Enter this command for the circuits to be cross-connected in the configuration mode for the circuit; namely, one of the following: ATM child circuit configuration mode, ATM PVC configuration mode, dot1q child circuit configuration mode, or dot1q PVC configuration mode. Do not use the **`bind interface`** command with crossed-connected circuits because they are bound instead by the **`xc`** command.
2.	Create or select the XC group for this cross-connection.	*xc-group (global)*	Enter this command in global configuration mode. Use the **`default`** keyword to select the default XC group.
3.	Create the cross-connection between the two circuits.	*xc*	Enter this command in XC group configuration mode. Encapsulation rules follow: • Cross-connecting circuits without child circuits: Do not specify the encapsulation of either circuit. • Cross-connecting a child circuit to a parent circuit: Specify the encapsulation of the child circuit. • Cross-connecting a child circuit to another child circuit: Specify the encapsulation of each child circuit. • Cross-connecting an ATM PVC to an 802.1Q PVC for interworking: Do not specify the encapsulation of either circuit; specify the **`interworking`** keyword. Please be aware of the following caution:

Caution!

Risk of data loss. To reduce the risk of data loss when configuring an interworking cross-connection between an ATM PVC with RFC 1483 routed encapsulation and an 802.1Q PVC, observe the following guidelines:

(1) To prevent long service interruption, the IPv4 device attached to the 802.1Q PVC must be configured with a low ARP time-out value. We recommend three minutes. (2) If the IP address changes for the IPv4 device attached to the ATM PVC, the administrator must ensure that the ARP cache on the SmartEdge router is cleared by using the **`clear arp-cache`** command (in exec mode) with the **`interworking`** keyword, for the IP address change to take effect.

2.3 Cross-Connection Operations

To monitor the cross-connections between circuits, perform the task listed in Table 2. Enter the **`show`** command in any mode.

Table 2 Cross-Connection Operations

Task	Root Command
Display bypass information.	*show bypass*

3 Configuration Examples

This section includes the following partial examples; only the commands to configure the cross-connections are included:

- Cross-Connected ATM PVCs
- Cross-Connected Child Circuits
- Cross-Connected Parent Circuit with Child Circuit
- Cross-Connected Circuits for Interworking

3.1 Cross-Connected ATM PVCs

The following example shows how to create two ATM PVCs on an ATM OC port with an existing profile, **ubr**, and encapsulated with **raw** mode, and cross-connect them, placing the cross-connection in the `xc-atm group:`

```
[local]Redback(config)#port atm 3/1
[local]Redback(config-atm-oc)#atm pvc 0 32 profile ubr encapsulation raw
[local]Redback(config-atm-pvc)#bind bypass
[local]Redback(config-atm-pvc)#exit
[local]Redback(config-atm-oc)#exit
!
[local]Redback(config)#port atm 4/2
[local]Redback(config-atm-oc)#atm pvc 1 55 profile ubr encapsulation raw
[local]Redback(config-atm-pvc)#bind bypass
!
[local]Redback(config)#xc-group xc-atm
[local]Redback(config-xc-group)#xc 3/1 vpi-vci 0 32 to 4/2 vpi-vci 1 55
```

Note: Fragmentation is not supported; the inbound and outbound IP packets are accepted or transmitted based on the size of the maximum transmission unit (MTU) of the inbound and outbound ports.

3.2 Cross-Connected Child Circuits

The following example shows how to create a cross-connection between two PPPoE child circuits on ATM PVCs on ATM OC ports with VPI **33** and VCIs **110** and **111** on the ATM ports **1** and **2** in slot **3**. Both PVCs use an ATM profile, **pf3**. Only the commands related to creating the cross-connection are shown.

```
[local]Redback#configure
!Create an ATM PVC with a PPPoE child circuit.
[local]Redback(config)#port atm 3/1
[local]Redback(config-atm-oc)#atm pvc 33 110 profile pf3 encapsulation multi
!Create the PPPoE child circuit on the ATM PVC.
[local]Redback(config-atm-pvc)#circuit protocol pppoe
[local]Redback(config-atm-child-proto)#bind bypass
[local]Redback(config-atm-child-proto)#end


!Create another ATM PVC with a PPPoE child circuit.
[local]Redback#configure
[local]Redback(config)#port atm 3/2
[local]Redback(config-atm-oc)#atm pvc 33 111 profile pf3 encapsulation multi
!Create the PPPoE child circuit on the ATM PVC.
[local]Redback(config-atm-pvc)#circuit protocol pppoe
[local]Redback(config-atm-child-proto)#bind bypass
[local]Redback(config-atm-child-proto)#end


!Cross-connect the PPPoE child circuits on the ATM PVCs
!and place the cross-connection in the xc-child group:
[local]Redback#configure
[local]Redback(config)#xc-group xc-child
[local]Redback(config-xc-group)#xc 3/1 vpi-vci 33 110 pppoe to
3/2 vpi-vci 33 111 pppoe
```

3.3 Cross-Connected Parent Circuit with Child Circuit

The following example shows how to create an ATM PVC with PPPoE encapsulation on an ATM OC port, an 802.1Q PVC with a child circuit, and cross-connect the ATM PVC with the PPPoE child circuit on the 802.1Q PVC:

```
!Create the ATM PVC
[local]Redback(config)#port atm 3/1
[local]Redback(config-atm-oc)#atm pvc 2 50 profile test encapsulation pppoe
[local]Redback(config-atm-pvc)#bind bypass
[local]Redback(config-atm-pvc)#exit
[local]Redback(config-atm-oc)#exit


!Create the 802.1Q PVC
[local]Redback(config)#port ethernet 2/1
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 1 encapsulation multi


!Create the PPPoE child circuit on the 802.1Q PVC.
[local]Redback(config-dot1q-pvc)#circuit protocol pppoe
[local]Redback(config-dot1q-child-proto)#bind bypass
[local]Redback(config-dot1q-child-proto)#exit
[local]Redback(config-dot1q-pvc)#exit
[local]Redback(config-port)#exit
[local]Redback(config)#


!Cross-connect the ATM PVC to the PPPoE child circuit on the 802.1Q PVC,
placing the cross-connection in the xc-mixed group
[local]Redback(config)#xc-group xc-mixed
[local]Redback(config-xc-group)#xc 3/1 vpi-vci 2 50 to 2/1 vlan-id 1 pppoe
```

3.4 Cross-Connected Circuits for Interworking

The following example shows how to create an ATM PVC with RFC 1483 routed encapsulation on an ATM OC port, an 802.1Q PVC, and an interworking

cross-connection between the two circuits. Only IPoE (IPv4) packets are forwarded across the connection:

```
!Create the ATM PVC and its IPoE circuit
[local]Redback(config)#port atm 3/1
[local]Redback(config-atm-oc)#atm pvc 3 110 profile test encapsulation route1483
[local]Redback(config-atm-pvc)#bind bypass
[local]Redback(config-atm-pvc)#exit
[local]Redback(config-atm-oc)#exit


!Create the 802.1Q PVC
[local]Redback(config)#port ethernet 2/1
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 5
[local]Redback(config-dot1q-pvc)#bind bypass
[local]Redback(config-dot1q-pvc)#exit
[local]Redback(config-port)#exit


!Cross-connect the ATM PVC to the 802.1Q PVC, placing the cross-connection in the
xc-interworking group
[local]Redback(config)#xc-group xc-interworking
[local]Redback(config-xc-group)#xc 3/1 vpi-vci 3 110 to 2/1 vlan-id 5 interworking
```

The following example shows how to create two ATM PVCs with RFC 1483 routed encapsulation on an ATM OC port, a pair of 802.1Q PVCs within an 802.1Q tunnel, and the interworking cross-connections between the ATM circuits and the 802.1Q circuits:

```
!Create the ATM PVCs on slot 1, port 1
[local]Redback(config)#port atm 1/1
[local]Redback(config-atm-oc)#atm pvc 0 100 profile ubr encapsulation route1483
[local]Redback(config-atm-pvc)#bind bypass
[local]Redback(config-atm-pvc)#exit
[local]Redback(config-atm-oc)#atm pvc 0 200 profile ubr encapsulation route1483
[local]Redback(config-atm-pvc)#bind bypass
[local]Redback(config-atm-pvc)#exit
[local]Redback(config-atm-oc)#exit


!Create the first 802.1Q PVC within an 802.1Q tunnel
[local]Redback(config)#port ethernet 2/1
[local]Redback(config-port)#encapsulation dot1q
[local]Redback(config-port)#dot1q pvc 10 encapsulation 1qtunnel
[local]Redback(config-dot1q-pvc)#exit
[local]Redback(config-port)#dot1q pvc 10:1
[local]Redback(config-dot1q-pvc)#bind bypass
[local]Redback(config-dot1q-pvc)#exit


!Create the second 802.1Q PVC within the 802.1Q tunnel
[local]Redback(config-port)#dot1q pvc 10:2 encapsulation multi
[local]Redback(config-dot1q-pvc)#bind bypass
[local]Redback(config-dot1q-pvc)#exit
[local]Redback(config-port)#exit


!Cross-connect the ATM PVCs to the 802.1Q PVCs, placing the
cross-connections in the xc-interworking group
[local]Redback(config)#xc-group xc-interworking
[local]Redback(config-xc-group)#xc 1/1 vpi-vci 0 100 to 2/1 vlan-id 10:1
interworking
[local]Redback(config-xc-group)#xc 1/1 vpi-vci 0 200 to 2/1 vlan-id 10:2
interworking
```