

Configuring Bridged Ethernet

18

This chapter describes the bridged Ethernet feature and describes how to configure bridged Ethernet.

The ERX system also supports bridged Ethernet on dynamic interfaces, which is described in *Chapter 21, Configuring Dynamic Interfaces*.

Topic	Page
Overview	18-1
References	18-3
Configuring Bridged Ethernet	18-4
Monitoring Bridged Ethernet	18-10

Overview

Bridged Ethernet allows multiple upper-layer interface types (IP, PPPoE, and CBF) to be simultaneously multiplexed over the same interface. You can set up the system to either terminate interfaces and route data or to pass data transparently through the system to another terminating device. This capability supports multiple client devices that use both IP and PPPoE encapsulation over an Ethernet LAN.

Connection-Based Forwarding

The CBF feature allows the system to pass data without terminating the layer 2 protocol. CBF provides a simple mechanism for forwarding traffic directly between ERX interfaces. CBF is a method of forwarding frames in which forwarding decisions are made using only the identity of the ingress interface. No part of a packet's contents is used to determine how a packet should be forwarded.

CBF Interfaces

Connection-based forwarding uses dedicated interfaces, called CBF interfaces. CBF interfaces used in connection-based forwarding are analogous to IP interfaces used in IP routing.

CBF Connections

CBF forwards frames along connections configured between pairs of CBF interfaces. Frames received on a connection's ingress interface are forwarded directly to that connection's egress interface. The packet is sent from ingress to egress interfaces as the raw payload received from the layer 2 interface. Once you create ingress and egress CBF interfaces, you then create a CBF connection to join the two interfaces.

Bridged Ethernet Application

Figure 18-1 and Figure 18-2 show an example of a client computer using IP/PPP/PPPoE and an Internet gaming system running IP, connecting to the ERX system over the same ATM PVC. The client computer and gaming system can connect to an ERX system via an xDSL modem over a single ATM PVC, and the system can forward these two data streams independently. When the system receives the two data streams, it uses the Ethertype contained in the bridged Ethernet header to select which upper interface (IP, PPPoE, or CBF) receives the frame.

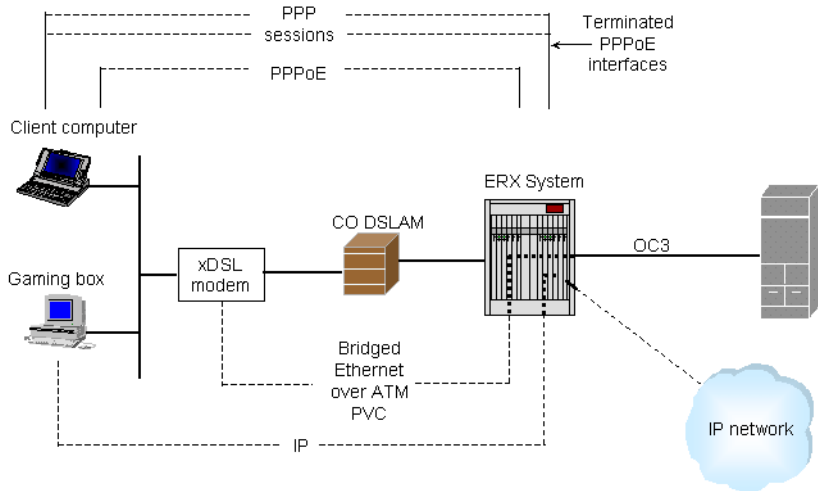


Figure 18-1 Bridged Ethernet topology with the system terminating and routing traffic

In Figure 18-1, IP and PPPoE interfaces are configured so that the non-PPPoE IP traffic is received by the IP interface, and the IP/PPP/PPPoE traffic is received by the PPPoE interface. Since the system receives these data streams on different IP interfaces, they may be routed independently.

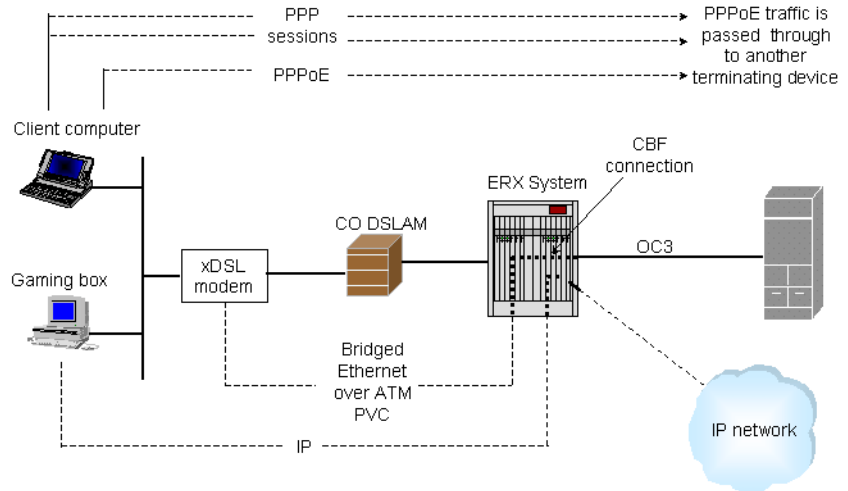


Figure 18-2 Bridged Ethernet topology with the system transparently passing PPPoE traffic to another terminating device

In Figure 18-2, IP and CBF interfaces are configured so that non-PPPoE IP traffic is received by the IP interface, and IP/PPP/PPPoE traffic is received by the ingress CBF interface. Here, the IP traffic is routed, and the PPPoE traffic is passed through to the egress CBF interface that the CBF connection points to.

References

Bridged Ethernet supports the following RFCs:

- RFC 2684 – Multiprotocol Encapsulation over ATM Adaptation Layer 5 (September 1999)
- RFC 826 – An Ethernet Address Resolution Protocol (November 1982)

Configuring Bridged Ethernet

This section shows how to configure IP with PPPoE terminated at the ERX system and IP with PPPoE passed through the ERX system. With each step, an illustration shows how the system is building the interface columns.

Configuring IP with PPPoE Terminated at the System

This section shows how to create IP with PPPoE interfaces that terminate the connection and route the data received on the PVC, as shown in the example in Figure 18-1. To create a terminated PVC:

- 1 Create an ATM1483 subinterface and associated PVC.

```
host1(config)#interface atm 9/1.1 point-to-point
host1(config-subif)#atm pvc 1 0 32 aal5snap 0 0 0
```

ATM 1483 9/1.1

- 2 Encapsulate the ATM1483 subinterface with bridged Ethernet. Note that the use of the **encapsulation** keyword implies that the bridged Ethernet interface is the only interface stacked directly above the ATM1483 subinterface. As a result, the bridged Ethernet interface cannot have a peer interface stacked above the same lower-layer interface.

```
host1(config-subif)#encapsulation bridge1483
```

Bridged Ethernet

ATM 1483 9/1.1

- 3 Create a PPPoE major interface over the bridged Ethernet interface. Note that this command does not use the **encapsulation** keyword.

```
host1(config-subif)#pppoe
```

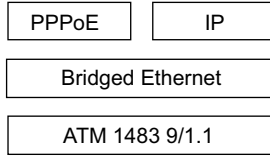
PPPoE

Bridged Ethernet

ATM 1483 9/1.1

- 4 Create an IP interface over the bridged Ethernet interface as a peer to the PPPoE interface.

```
host1(config-subif)#ip address 160.1.0.1 255.255.255.0
```



- (Optional) Set up the system to validate MAC addresses on the IP interface.

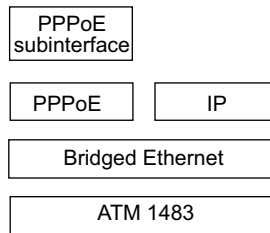
```
host1(config-subif)#ip mac-validate strict
```

- 5 Exit the subinterface context.

```
host1(config-subif)#exit
```

- 6 Create a PPPoE subinterface over the major interface.

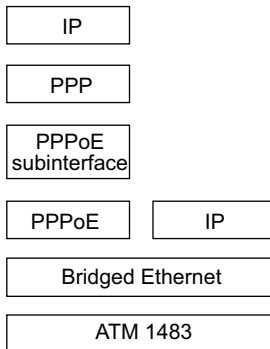
```
host1(config)#pppoe subinterface atm 9/1.1.1
```



- 7 Configure PPP encapsulation over the PPPoE subinterface, and IP interface over the PPP interface.

```
host1(config-subif)#encapsulation ppp
```

```
host1(config-subif)#ip address 160.1.1.1 255.255.255.0
```



Configuring IP with PPPoE Passed Through the System

This section shows how to create IP with a CBF interface that passes the PPPoE data received through the system to another terminating device, as shown in the example in Figure 18-2. To create a configuration that transparently passes data:

- 1 Create the ATM1483 subinterface and associated PVC.

```
host1(config)#interface atm 9/1.1 point-to-point
host1(config-subif)#atm pvc 1 0 32 aal5snap 0 0 0
```

ATM 1483 (9/1.1)

- 2 Encapsulate the ATM1483 subinterface with bridged Ethernet. Note that the use of the **encapsulation** keyword implies that the bridged Ethernet interface is the only interface stacked directly above the ATM1483 subinterface.

```
host1(config-subif)#encapsulation bridge1483
```

Bridged Ethernet

ATM 1483 (9/1.1)

- 3 Create a CBF interface over the bridged Ethernet interface.

```
host1(config-subif)#cbf
```

CBF

Bridged Ethernet

ATM 1483 (9/1.1)

- 4 Exit the subinterface context.

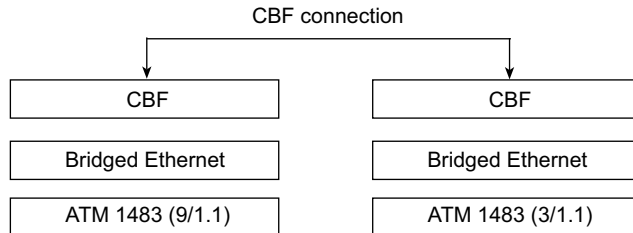
```
host1(config-subif)#exit
```

- 5 Create a second ATM interface that will also have a bridged Ethernet and CBF interface.

```
host1(config)#interface atm 3/1.1 point-to-point
host1(config-subif)#atm pvc 1 0 32 aal5snap 0 0 0
host1(config-subif)#encapsulation bridge1483
host(config-subif)#cbf
host(config-subif)#exit
```

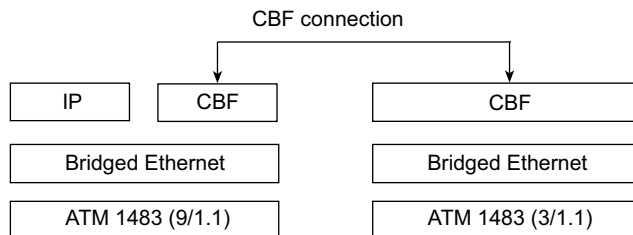
- 6 Create a CBF connection between the two ATM interfaces. Once this connection is established, all traffic will travel over the CBF connection.

```
host1(config)#cbf connection atm 9/1.1 atm 3/1.1
```



- 7 Return to the ATM 9/1.1 context, and add an IP interface as a peer to the CBF interface. Once this IP interface is added, all IP traffic will be handled by the IP interface. All other traffic will be sent to the CBF interface.

```
host1(config)#interface atm 9/1.1 point-to-point
host1(config-subif)#ip address 160.1.0.1 255.255.255.0
```



- (Optional) Set up the system to validate MAC addresses on the IP interface.

```
host1(config-subif)#ip mac-validate strict
```

atm pvc

- Use to configure a PVC on an ATM interface. Specify one of the following encapsulation types:
 - › **aal5snap** – specifies a logical link control (LLC) encapsulated circuit; LLC/Subnetwork Access Protocol (LLC/SNAP) precedes the protocol datagram.
 - › **aal5mux ip** – specifies a VC multiplexed circuit. This option is used for IP only.

- Example

```
host1(config-subif)#atm pvc 1 0 32 aal5snap 0 0 0
```

- Use the **no** version to remove the specified PVC.

cbf

- Use to create a CBF interface over the interface that you are configuring.
- Example

```
host1(config-subif)#cbf
```
- Use the **no** version to remove the CBF interface.

cbf connection

- Use to create a CBF connection between two CBF interfaces.
- Example

```
host1(config)#cbf connection atm 9/1.1 atm 3/1.1
```
- Use the **no** version to remove an existing connection.

encapsulation bridge1483

- Use to configure bridged Ethernet as the encapsulation method on an interface.
- Example

```
host1(config-subif)#encapsulation bridge1483
```
- Use the **no** version to remove bridged Ethernet as the encapsulation method on the interface.

encapsulation ppp

- Use to configure PPP as the encapsulation method for an interface.
- Example

```
host1(config-subif)#encapsulation ppp
```
- Use the **no** version to remove PPP as the encapsulation method on the interface.

interface atm

- Use to configure an ATM interface or subinterface type in the *slot/port.subinterface* format:
 - › *slot* – system chassis slot
 - › *port* – I/O module port
 - › *subinterface* – number of the subinterface in the range 1–4294967293
- Example

```
host1(config)#interface atm 9/1.1 point-to-point
```
- Use the **no** version to remove the interface or subinterface.

ip address

- Use to set an IP address for the interface.
- Note that you cannot add more than one IP address to bridged Ethernet interfaces.

- Example


```
host1(config-subif)#ip address 160.1.0.1 255.255.255.0
```
- Use the **no** version to remove the IP address.

ip mac-validate

- Use to enable or disable MAC address validation on a per interface basis.
- Use the **strict** keyword to prevent transmission of IP packets that do not reside in the validation table.
- Use the **loose** keyword to allow IP packets to pass through even though the packets do not have entries in the validation table. Only packets that have matching IP–MAC pair entries in the table are validated.
- There is no default for this command.
- Example

```
host1(config-subif)#ip mac-validate strict
```

- Use the **no** version to disable the command.



Note: For more information about MAC address validation, see *MAC Address Validation in ERX Routing Protocols Configuration Guide, Vol. 1, Chapter 2, Configuring IP*.

pppoe

- Use to create a PPPoE major interface.
- Example

```
host1(config-subif)#pppoe
```

- Use the **no** version to remove the PPPoE major interface.

pppoe subinterface atm

- Use to create a PPPoE subinterface on an ATM interface in the *slot/port.atmSubinterface.pppoeSubinterface* format:
 - › *slot* – system chassis slot
 - › *port* – I/O module port
 - › *atmSubinterface* – number of the ATM subinterface in the range 1–2147483647
 - › *pppoeSubinterface* – number of the PPPoE subinterface in the range 1–2147483647
- Example

```
host1(config)#pppoe subinterface atm 9/1.1.1
```

- Use the **no** version to remove the PPPoE subinterface.

Backward-Compatible Configuration

Before software release 3.4, you could configure a PPPoE major interface directly over ATM1483 only. The system still supports this stacking and configuration method for PPPoE. Although the older and newer interface

stacks are different, they behave the same in terms of frame encapsulation and handling. As a result, an interface created using the older stacking will interoperate with an interface using the new stacking. Note, however, that the previous command syntax (**encapsulation pppoe**) cannot be used when a bridged Ethernet interface already exists, because it is intended to produce the old stacking for PPPoE/ATM1483.

- 1 Create the ATM1483 subinterface and associated PVC:

```
host1(config)#interface atm 9/1.1 point-to-point  
host1(config-subif)#atm pvc 1 0 32 aa15snap 0 0 0
```

- 2 Create a PPPoE major interface over the bridged Ethernet interface. Note that since this command uses the **encapsulation** keyword, it will fail if a bridged Ethernet interface was already created over the ATM1483 interface using the new syntax.

```
host1(config-subif)#encapsulation pppoe
```

- 3 Create a PPPoE subinterface over the major interface. Because PPPoE is the only top layer protocol in the stack, there is no need to use **pppoe** to identify the subinterface type (it is implied).

```
host1(config)#interface atm 9/1.1.1
```

- 4 Configure the PPP encapsulation over the PPPoE subinterface, and IP interface over the PPP interface.

```
host1(config-subif)#encapsulation ppp  
host1(config-subif)#ip address 160.1.1.1 255.255.255.0
```

Monitoring Bridged Ethernet

To monitor bridged Ethernet, use the **show configuration** command. Regardless of the CLI method used to configure the system (the backward compatible configuration or the new configuration style), the **show configuration** command produces output consistent with the interface stack that was created.

Because bridged Ethernet interfaces can have multiple upper interfaces, they are not bound to a specific virtual router. If you enter **show configuration** without a VR filter, a bridged Ethernet interface with multiple upper interfaces is displayed in the default VR configuration and again in any VR in which the bridged Ethernet interface has an upper interface that is bound to that VR.

To monitor the MAC address validation configuration, use the **show ip mac-validate** command.

show ip mac-validate

- Use to display the status of the MAC address validation on the physical interface.
- Field descriptions
 - › *interfaceSpecifier* – interface type slot/port
 - › Keyword assigned to interface – Strict or Loose
 - › Address – IP address of the entry
 - › Hardware Addr – physical (MAC) address of the entry
- Example

```
host1:vr1#show ip mac-validate interface atm 8/0.1  
ATM8/0.1: Strict
```

Address	Hardware Addr
180.1.0.2	0000.1111.2222

