Configuring Frame Relay

Frame Relay was conceived as a protocol for use over serial interfaces and was designed for networks with large T1 installations. This chapter describes the tasks for configuring Frame Relay on the router. For a complete description of the commands mentioned in this chapter, refer to the “Frame Relay Commands” chapter in the *Router Products Command Reference* publication. For historical background and a technical overview of Frame Relay, see the *Internetworking Technology Overview* publication.

Cisco’s Implementation of Frame Relay

Cisco’s Frame Relay implementation currently supports routing on IP, DECnet, AppleTalk, Xerox Network Service, Novell IPX, ISO CLNS, Banyan VINES, and transparent bridging.

The Frame Relay software provides the following capabilities:

- Support for the three generally implemented specifications of Frame Relay Local Management Interfaces (LMIs):
    - The *Frame Relay Interface* joint specification produced by Northern Telecom, Digital Equipment Corporation, StrataCom, and Cisco Systems
    - The ANSI-adopted Frame Relay signal specification, T1.617 Annex D
    - The International Telecommunication Union Telecommunication Standardization Sector (ITU-T)-adopted Frame Relay signal specification, Q.933 Annex A

Note The ITU-T carries out the functions of the former Consultative Committee for International Telegraph and Telephone (CCITT).

- Conformity to ITU-T I-series (ISDN) recommendation as I122, “Framework for Additional Packet Mode Bearer Services.”
    - The ANSI-adopted Frame Relay encapsulation specification, T1.618
    - The ITU-T-adopted Frame Relay encapsulation specification, Q.922 Annex A
- Conformity to Internet Engineering Task Force (IETF) encapsulation in accordance with RFC 1294, except bridging.

- Support for a keepalive mechanism, a multicast group, and a status message, as follows:
    - The keepalive mechanism provides an exchange of information between the network server and the switch to verify that data is flowing.
    - The multicast mechanism provides the network server with its local data link connection identifier (DLCI) and the multicast DLCI. This feature is specific to our implementation of the Frame Relay joint specification.
    - The status mechanism provides an ongoing status report on the DLCIs known by the switch.
- Transmission of congestion information from Frame Relay to DECnet Phase IV and CLNS. This mechanism promotes Forward Explicit Congestion Notification (FECN) bits from the Frame Relay layer to upper-layer protocols after checking for the FECN bit on the incoming DLCI. Use this Frame Relay congestion information to adjust the sending rates of end hosts. FECN-bit promotion is enabled by default on any interface using Frame Relay encapsulation. No configuration is required.
- Support for Frame Relay Inverse Address Resolution Protocol (InvARP) as described in RFC 1293 for the AppleTalk, Banyan VINES, DECnet, IP, and IPX protocols, as well as native Hello packets for DECnet, CLNP, and Banyan VINES. It allows a router running Frame Relay to discover the protocol address of a device associated with the virtual circuit.
- Support for Frame Relay switching, whereby packets are switched based on the DLCI (a Frame Relay equivalent of a MAC-level address). Routers are configured as a hybrid DTE switch or pure Frame Relay DCE access node in the Frame Relay network. Cisco's implementation of Frame Relay switching allows the following configurations:
    - Switching over an IP tunnel
    - Network to Network Interface (NNI) to other Frame Relay switches
    - Local serial-to-serial switching
- Support for *subinterfaces* associated with a physical interface. The software groups one or more permanent virtual circuits under separate subinterfaces, which in turn are located under a single physical interface. See the "Configuring Interfaces" chapter in the *Router Products Configuration Guide* for more information about configuring subinterfaces on serial interfaces running Frame Relay encapsulation. Also see the sections "Associate a DLCI with an Interface" and "Frame Relay Configuration Examples" later in this chapter.
- Support of the Frame Relay DTE MIB specified in RFC 1315. However, the error table is not implemented. To use the Frame-Relay MIB, refer to your MIB publications.

Frame Relay Hardware Requirements

One of the following hardware configurations is possible for Frame Relay connections:

- Routers can connect directly to the Frame Relay switch.
- Routers can connect directly to a Channel Service Unit/Digital Service Unit (CSU/DSU) first, and the CSU/DSU is connected to a remote Frame Relay switch.

Note A Frame Relay network is not required to support only routers that are connected directly or only routers connected via CSU/DSUs. Within a network, some routers can connect to a Frame Relay switch through a direct connection and others through connections via CSU/DSUs. However, a single router interface configured for Frame Relay can be only one or the other.

The CSU/DSU converts V.35 or RS-449 signals to the properly coded T1 transmission signal for successful reception by the Frame Relay network. Figure 9-1 illustrates the connections between the different components.

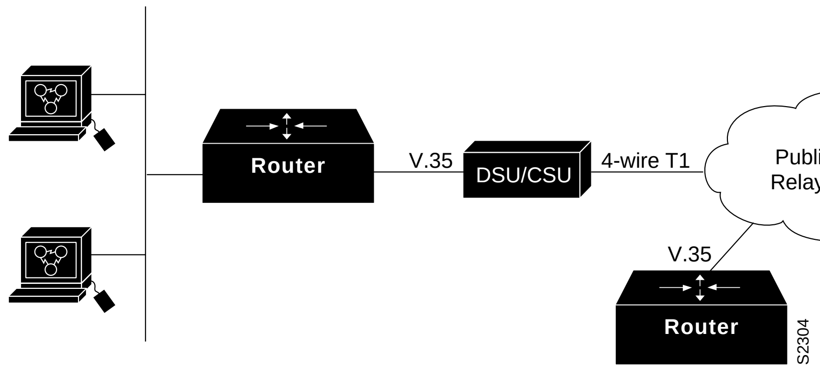

Table 9-1 Typical Frame Relay Configuration

The Frame Relay interface actually consists of one physical connection between the network server and the switch that provides the service. This single physical connection provides direct connectivity to each device on a network, such as a StrataCom FastPacket wide area network.

Frame Relay Configuration Task List

There are required, basic steps you must follow to enable Frame Relay for your network. In addition, you can customize Frame Relay for your particular network needs, set local and multicast DLCIs in test environments, and monitor Frame Relay connections. The following sections outline these tasks. You must perform the tasks in the first section.

- Enable Frame Relay on an Interface
- Customize Your Frame Relay Network
- Configure Frame Relay in a Test Environment
- Monitor the Frame Relay Connections

The following sections describe these tasks. See the examples at the end of this chapter for ideas of how to configure Frame Relay on your network. See the “Frame Relay Commands” chapter in the *Router Products Command Reference* publication for information about the commands listed in the tasks.

Enable Frame Relay on an Interface

You must perform the tasks in the following sections to enable Frame Relay:

- Set Frame Relay Encapsulation
- Establish Mapping

Set Frame Relay Encapsulation

To set Frame Relay encapsulation at the interface level, perform the following task in interface configuration mode:

Task	Command
Enable Frame Relay and specify the encapsulation method.	**encapsulation frame-relay** [**ietf**]

Frame Relay supports encapsulation of all supported protocols except bridging in conformance with RFC 1294, allowing interoperability between multiple vendors. Use the IETF form of Frame Relay encapsulation if your router is connected to another vendor's equipment across a Frame Relay network. IETF encapsulation is supported at either the interface level or on a per-DLCI (map entry) basis.

For an example of how to enable Frame Relay and set the encapsulation method, see the sections "Configurations Using IETF Encapsulation Example" and "Two Routers in Static Mode Example" later in this chapter. Also see the "Configuring Interfaces" chapter in you want to configure subinterfaces on serial interfaces running Frame Relay encapsulation.

Establish Mapping

The Frame Relay map tells the network server how to get from a specific protocol and address pair to the correct local data link connection identifier (DLCI). To establish mapping according to your network needs, perform one of the following tasks in interface configuration mode:

Task	Command
Define the mapping between a supported protocol address and the DLCI used to connect to the address.	**frame-relay map** *protocol protocol-address DLCI* [**broadcast**] [**ietf**] [**cisco**]
Define the mapping between an address and the DLCI used to connect, using ISO CLNS protocol.	**frame-relay map clns** *dlci* [**broadcast**]
Define the mapping between an address and the DLCI used to connect to a bridge.	**frame-relay map bridge** *dlci* [**broadcast**]

The supported protocols and bridging feature with the corresponding keywords to enable them are as follows:

- IP—**ip**
- DECnet—**decnet**
- AppleTalk—**appletalk**
- XNS—**xns**
- Novell IPX—**ipx**
- VINES—**vines**
- ISO CLNS—**clns**
- Bridging—**bridge**

The configuration for the Open Shortest Path First (OSPF) protocol can be greatly simplified by adding the optional **broadcast** keyword when doing this task. See the **frame-relay map** description in the *Router Products Command Reference* publication and the examples at the end of this chapter for more information about using the **broadcast** keyword.

For an example of how to establish mapping, see the sections “Two Routers in Static Mode Example,” “Routing DECnet Packets Example,” and “Routing IPX Packets Example” later in this chapter.

Customize Your Frame Relay Network

Perform the tasks in the following sections to customize Frame Relay:

- Configure Frame Relay Switching
- Configure the LMI
- Select Frame Relay Inverse ARP
- Associate a DLCI with a Subinterface

Configure Frame Relay Switching

Frame Relay switching is a means of switching packets based upon the DLCI, which can be looked upon as the Frame Relay equivalent of a MAC address. The switching is performed by configuring your router as a Frame Relay network. There are two parts to a Frame Relay network: a Frame Relay DTE (the router) and a Frame Relay DCE switch. Figure 9-1 illustrates this concept.

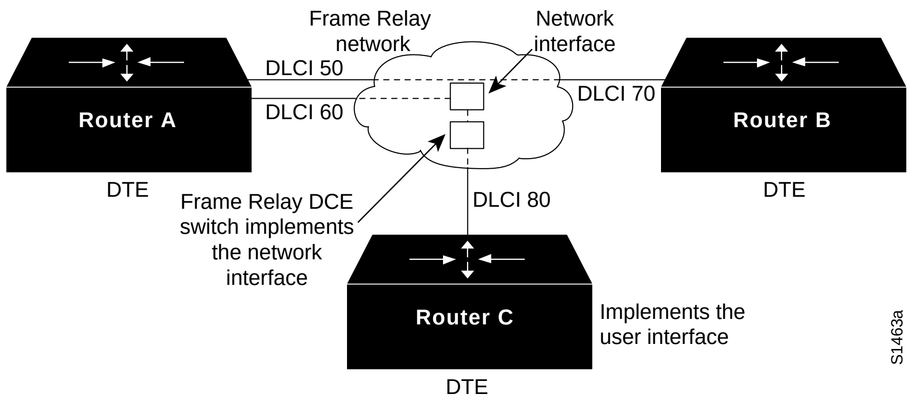

Figure 9-1 Frame Relay Switched Network

In Figure 9-1, Routers A, B, and C are Frame Relay DTEs connected to each other via a Frame Relay network. Our implementation of Frame Relay switching allows our routers to be used as depicted in this Frame Relay network.

Perform the tasks in the following sections, as necessary, to configure Frame Relay switching:

- Enable Frame Relay Switching
- Configure a Frame Relay DTE Device, DCE Switch, or NNI Support
- Specify the Static Route

These tasks are described in the following sections.

Enable Frame Relay Switching

You must enable packet switching before you can configure it on a Frame Relay DTE, DCE, or with Network to Network Interface (NNI) support. Do so by performing the following task in global configuration mode before configuring the switch type:

Task	Command
Enable Frame Relay switching.	**frame-relay switching**

For an example of how to enable Frame Relay switching, see the switching examples later in this chapter.

Configure a Frame Relay DTE Device, DCE Switch, or NNI Support

You can configure your router as a DTE device, DCE switch, or as a switch connected to a switch to support NNI connections. (DCE is the default.) To do so, perform the following task in interface configuration mode:

Task	Command
Configure a Frame Relay DTE device or DCE switch.	**frame-relay intf-type** [**dce** | **dte** | **nni**]

For an example of how to configure a DTE device or DCE switch, see the section “Hybrid DTE/DCE PVC Switching Example” later in this chapter.

For an example of how to configure NNI support, see the section “Configuring a Pure Frame Relay DCE Example” later in this chapter.

Specify the Static Route

You must specify a static route for PVC switching. To do so, perform the following task in interface configuration mode:

Task	Command
Specify the static route for PVC switching.	**frame-relay route** *in-dlci out-interface out-dlci*

For an example of how to specify a static route, see the section “Configuring a Pure Frame Relay DCE Example” later in this chapter.

Configure the LMI

Our Frame Relay software supports the industry-accepted standards for addressing the Local Management Interface (LMI), including the Cisco specification. You can enable the following LMI features:

- Set the LMI type, either ANSI, CCITT, or Cisco.
- Set the LMI keepalive interval.
- Set LMI polling intervals, timer intervals, and error thresholds; parameters exist for both DTE and DCE device types.

Set the LMI Type

You can set one of three types of LMIs on our router: ANSI T1.617 Annex D, Cisco, and ITU-T Q.933 Annex A. To do so, perform the following task in interface configuration mode:

Task	Command
Set the LMI type.	**frame-relay lmi-type** {**ansi** | **cisco** | **q933a**}

For an example of how to set the LMI type, see the section “Configuring a Pure Frame Relay DCE Example” later in this chapter.

Set the LMI Keepalive Interval

A keepalive interval must be set to enable LMI. By default, this interval is ten seconds and, per the LMI protocol, must be less than the corresponding interval on the switch. To set the keepalive interval, perform the following task in interface configuration mode:

Task	Command
Set the keepalive interval.	**frame-relay keepalive** *number*
Turn off keepalives on networks without an LMI.	**no frame-relay keepalive**

This command has the same effect as the **keepalive** interface configuration command.

The keepalive interval cannot be enabled when the LMI is disabled; they go together. For an example of how to specify an LMI keepalive interval, see the section “Two Routers in Static Mode Example” later in this chapter.

Set the LMI Polling and Timer Intervals

You can set various counters, intervals, and thresholds to fine-tune the operation of your LMI DTE and DCE devices. See the following table for the tasks that you can perform. See the "Frame Relay Commands" chapter in the *Router Products Command Reference* publication for details about commands used to set the polling and timing intervals. Set these intervals by performing one or more of the following tasks in interface configuration mode:

Task	Command
Set the DCE and NNI error threshold.	**frame-relay lmi-n392dce** *threshold*
Set the DCE and NNI monitored events count.	**frame-relay lmi-n393dce** *events*
Set the polling verification timer on a DCE or NNI interface.	**frame-relay lmi-t392dce** *timer*
Set a full status polling interval on a DTE or NNI interface.	**frame-relay lmi-n391dte** *keep-exchanges*
Set the DTE or NNI error threshold.	**frame-relay lmi-n392dte** *threshold*
Set the DTE and NNI monitored events count.	**frame-relay lmi-n393dte** *events*

Select Frame Relay Inverse ARP

Frame Relay Inverse ARP is a method of building dynamic routes in Frame Relay networks running AppleTalk, Banyan VINES, DECnet, IP, Novell IPX, and XNS. Inverse ARP allows the router to discover the protocol address of a device associated with the virtual circuit. Inverse ARP is used instead of the **frame-relay map** command, which allows you to define the mappings between a specific protocol and address and a specific DLCI (see the section "Establish Mapping" earlier in this chapter for more information).

Inverse ARP is enabled by default. Configure Inverse ARP if you want to configure an interface for multipoint communication that was previously configured for point-to-point. You would not need to select Inverse ARP if you have a point-to-point interface, because there is only a single destination and discovery is not required.

To select Inverse ARP, perform the following task in interface configuration mode:

Task	Command
Select Frame Relay Inverse ARP.	**frame-relay inverse-arp** *protocol dlci*

Define Subinterfaces

Subinterfaces solve many of the problems seen in protocols such as AppleTalk that have split horizon enabled and no capability to disable it. However, not all protocols support subinterfaces. See the "Configuring Interfaces" chapter in the *Router Products Configuration Guide* for information for a list of protocols that support subinterfaces. For more information about split horizon, refer to the "Configuring IP Routing Protocols" in the *Router Products Configuration Guide*.

You can configure subinterfaces for multipoint or point-to-point communication. Point-to-point is the default. To configure an interface for multipoint or point-to-point communication, you must first define an interface in global configuration mode. After defining an interface, you can define a subinterface for that interface by performing the following task in interface configuration mode:

Task	Command
Define a subinterface.	**interface** *interface-type subinterface-number* [**multipoint** | **point-to-point**][1]

1. This command is documented in the "Interface Commands" chapter in the *Router Products Command Reference* publication.

Once you have defined the subinterface, you must perform one of the following tasks in interface configuration mode:

- Establish mapping.
- Select Frame-Relay Inverse ARP.
- Associate a DLCI with a subinterface.

If you define a subinterface for multipoint communication, you cannot use the **frame-relay-interface-dlci** command. If you define a subinterface for point-to-point communication, you cannot use the **frame-relay map** command. The **frame-relay inverse-arp** command is designed for use with an interface configured for multipoint communication and should not be used for a subinterface configured for point-to-point communication.

Note If you define a subinterface for point-to-point communication, you cannot reassign the same subinterface number to be used for multipoint communication without first rebooting the router.

For an example of how to define a subinterface, see the section "Transparent Bridging Using Subinterfaces Example" later in this chapter.

Associate a DLCI with a Subinterface

You must associate the Frame Relay DLCI with a subinterface to use subinterfaces in the Frame Relay network for point-to-point communication. If you associate a DLCI with a point-to-point subinterface, you cannot use the **frame-relay map** command.

To associate a DLCI with a subinterface, perform the following task in interface configuration mode:

Task	Command
Associate a DLCI with a subinterface.	**frame-relay interface-dlci** *dlci* [*option*]

For an example of how to associate a DLCI with a subinterface, see the section "Transparent Bridging Using Subinterfaces Example" later in this chapter.

Create a Broadcast Queue for an Interface

Very large Frame Relay networks might have performance problems when very many DLCIs terminate in a single router and the router must replicate routing updates and service advertising updates on each DLCI. The updates can consume access-link bandwidth and cause significant latency variations in user traffic; the updates can also consume interface buffers and lead to higher packet rate loss for both user data and routing updates.

To avoid such problems, you can create a special broadcast queue for an interface. The broadcast queue is managed independently of the normal interface queue, has its own buffers, and has a configurable size and service rate.

A broadcast queue is given a maximum transmission rate (throughput) limit measured in both bytes per second and packets per second. The queue is serviced to ensure that no more than this maximum is provided. The broadcast queue has priority when transmitting at a rate below the configured maximum, and hence has a guaranteed minimum bandwidth allocation. The two transmission rate limits are intended to avoid flooding the interface with broadcasts. The actual transmission rate limit in any second is the first of the two rate limits that is reached.

To create a broadcast queue, complete the following task in interface configuration mode:

Task	Command
Create a broadcast queue for an interface.	**frame-relay broadcast-queue** *size byte-rate packet-rate*

Configure TCP/IP Header Compression

TCP/IP header compression, as described by RFC 1144, is designed to improve the efficiency of bandwidth utilization over low-speed serial links. A typical TCP/IP packet includes a 40-byte datagram header. Once a connection is established, the header information is redundant and need not be repeated in every packet that is sent. By reconstructing a smaller header that identifies the connection and indicates the fields that changed and the amount of change, fewer bytes can be transmitted. The average compressed header is 10 bytes long.

For this algorithm to function, packets must arrive in order. If packets arrive out of order, the reconstruction will appear to create regular TCP/IP packets but the packets will not match the original. Because priority queuing changes the order in which packets are transmitted, enabling priority queueing on the interface is not recommended.

You can configure TCP/IP header compression in either of two ways, as described in the following sections:

- Configure an Individual IP Map for TCP/IP Header Compression
- Configure an Interface for TCP/IP Header Compression

The "Disable TCP/IP Header Compression" section describes how to disable this feature.

Note If you configure an interface with Cisco encapsulation and TCP/IP header compression, Frame Relay IP maps inherit the compression characteristics of the interface. However, if you configure the interface with IETF encapsulation, the interface cannot be configured for compression. Frame Relay maps will have to be configured individually to support TCP/IP header compression.

Configure an Individual IP Map for TCP/IP Header Compression

TCP/IP header compression requires Cisco encapsulation. If you need to have IETF encapsulation on an interface as a whole, you can still configure a specific IP map to use Cisco encapsulation and TCP header compression.

In addition, even if you configure the interface to perform TCP/IP header compression, you can still configure a specific IP map not to compress TCP/IP headers.

You can specify whether TCP/IP header compression is active or passive. Active compression subjects every outgoing packet to TCP/IP header compression. Passive compression subjects an outgoing TCP/IP packet to header compression only if the packet had a compressed TCP/IP header when it was received.

To configure an IP map to use Cisco encapsulation and TCP/IP header compression, perform the following task in interface configuration mode:

Task	Command
Configure an IP map to use Cisco encapsulation and TCP/IP header compression.	**frame-relay map ip** *ip-address dlci* [**broadcast**] **cisco tcp header-compression** {**active** | **passive**}

The default encapsulation is **cisco**.

Note An interface that is configured to support TCP/IP header compression cannot also support priority queuing or custom queuing.

For an example of how to configure TCP header compression on an IP map, see the "TCP/IP Header Compression Examples" section later in this chapter.

Configure an Interface for TCP/IP Header Compression

You can configure the interface with active or passive TCP/IP header compression. Active compression, the default, subjects all outgoing TCP/IP packets to header compression. Passive compression subjects an outgoing packet to header compression only if the packet had a compressed TCP/IP header when it was received on that interface.

To apply TCP/IP header compression to an interface, you must perform the following tasks in interface configuration mode:

Task	Command
Configure Cisco encapsulation on the interface.	**encapsulation frame-relay**
Enable TCP/IP header compression on the interface.	**frame-relay ip tcp header-compression** [**passive**]

Note If an interface configured with Cisco encapsulation is later configured with IETF encapsulation, all TCP/IP header compression characteristics are lost. To apply TCP/IP header compression over an interface configured with IETF encapsulation, you must configure individual IP maps, as described in the section "Configure an Individual IP Map for TCP/IP Header Compression."

For an example of how to configure TCP header compression on an interface, see the "TCP/IP Header Compression Examples" section later in this chapter.

Disable TCP/IP Header Compression

You can disable TCP/IP header compression by using either of two commands that have different effects, depending on whether Frame Relay IP maps have been explicitly configured for TCP/IP header compression or have inherited their compression characteristics from the interface.

Frame Relay IP maps that have explicitly configured TCP/IP header compression must also have TCP/IP header compression explicitly disabled.

To disable TCP/IP header compression, perform one of the following tasks in interface configuration mode:

Task	Command
Disable TCP header compression on all Frame Relay IP maps that are not explicitly configured for TCP header compression.	**no frame-relay ip tcp header-compression**
or	
Disable TCP header compression on a specified Frame Relay IP map.	**frame-relay map ip** *ip-address dlci* **nocompress**

For an example of how to turn off TCP header compression, see the section "Disabling TCP/IP Header Compression Examples."

Configure Discard Eligibility

You can specify which Frame Relay packets have low priority or low time sensitivity and will be the first to be dropped when a Frame Relay switch is congested. The mechanism that allows a Frame Relay switch to identify such packets is the discard eligibility (DE) bit.

This feature requires that the Frame Relay network be able to interpret the DE bit. Some networks take no action when the DE bit is set. Other networks use the DE bit to determine which packets to discard. The most desirable interpretation is to use the DE bit to determine which packets should be dropped first and also which packets have lower time sensitivity.

You can define DE lists that identify the characteristics of packets to be eligible for discarding, and you can also specify DE groups to identify the DLCI that is affected.

To define a DE list specifying the packets that can be dropped when the Frame Relay switch is congested, perform the following task in global configuration mode:

Task	Command
Define a DE list.	**frame-relay de-list** *list-number* **protocol** {*protocol* | *type number*} *characteristic*

You can specify DE lists based on the protocol or the interface, and on characteristics such as fragmentation of the packet, a specific TCP or UDP port, an access list number, or packet size. See the **frame-relay de-list** command for arguments and other information.

To define a DE group specifying the DE list and DLCI affected, perform the following task in interface configuration mode:

Task	Command
Define a DE group.	**frame-relay de-group** *group-number dlci*

Configure Frame Relay in a Test Environment

Perform the following tasks only if you are configuring Frame Relay in a test environment:

- Set the Local DLCI
- Set the DLCI for Multicasts

Set the Local DLCI

You can set a local DLCI in a test environment. This feature is provided mainly to allow testing of the Frame Relay encapsulation in a setting where two routers are connected back to back. This command is not required in a live Frame Relay network. Its use allows the source local DLCI to be set for use when the LMI is not supported. To set the local DLCI, perform the following task in interface configuration mode:

Task	Command
Set a local DLCI.	**frame-relay local-dlci** *number*

If LMI is supported and the multicast information element is present, the network server sets its local DLCI based on information provided via the LMI.

Set the DLCI for Multicasts

You can specify a DLCI for multicasts in a test environment. This feature is provided mainly to allow testing of the Frame Relay encapsulation in a setting where two routers are connected back to back. This task is not required in a live Frame Relay network. Its use allows network transmissions (packets) sent to a multicast DLCI to be delivered to all network servers defined as members of the multicast group. To set the DLCI for multicasts, perform the following task in interface configuration mode:

Task	Command
Specify a DLCI for multicasts in a test environment.	**frame-relay multicast-dlci** *number*

Monitor the Frame Relay Connections

To monitor Frame Relay connections, perform any of the following tasks in EXEC mode:

Task	Command
Clear dynamically created Frame Relay maps, which are created by the use of inverse ARP.	**clear frame-relay-inarp**
Display information about the Frame Relay DLCI and the LMI.	**show interfaces serial** *number*
Display LMI statistics.	**show frame-relay lmi** [*type number*]
Display the current Frame Relay map entries.	**show frame-relay map**
Display PVC statistics.	**show frame-relay pvc** [*type number* [*dlci*]]
Display configured static routes.	**show frame-relay route**
Display Frame Relay traffic statistics.	**show frame-relay traffic**

Frame Relay Configuration Examples

This section provides examples of Frame Relay configurations. It includes the following examples:

- Configurations Using IETF Encapsulation Example
- TCP/IP Header Compression Examples
- Disabling TCP/IP Header Compression Examples
- Two Routers in Static Mode Example
- Routing DECnet Packets Example
- Routing IPX Packets Example
- Configuration Providing Backward Compatibility Example
- Netbooting over Frame Relay Example
- Transparent Bridging Using Subinterfaces Example
- PVC Switching Configuration Example
- Configuring a Pure Frame Relay DCE Example
- Hybrid DTE/DCE PVC Switching Example
- Switching over an IP Tunnel Example

Configurations Using IETF Encapsulation Example

The first example that follows sets IETF encapsulation at the interface level. The second example sets IETF encapsulation on a per-DLCI basis. In the first example, the keyword **ietf** sets the default encapsulation method for all maps to IETF.

```
encapsulation frame-relay IETF
frame-relay map ip 131.108.123.2 48 broadcast
frame-relay map ip 131.108.123.3 49 broadcast
```

In the following example, IETF encapsulation is configured on a per-DLCI basis. This configuration has the same result as the configuration in the first example.

```
encapsulation frame-relay
frame-relay map ip 131.108.123.2 48 broadcast ietf
frame-relay map ip 131.108.123.3 49 broadcast ietf
```

TCP/IP Header Compression Examples

The following examples show various combinations of TCP/IP header compression and encapsulation characteristics on the interface and the effect on the inheritance of those characteristics on a Frame Relay IP map.

IP Map with Inherited TCP/IP Header Compression Example

The following example shows an interface configured for TCP/IP header compression and an IP map that inherits the compression characteristics. Note that the Frame Relay IP map is not explicitly configured for header compression.

```
interface serial 1
encapsulation frame-relay
ip address 131.108.177.178 255.255.255.0
frame-relay map ip 131.108.177.177 177 broadcast
frame-relay ip tcp header-compression passive
```

Use of the **show frame-relay map** command will display the resulting compression and encapsulation characteristics; the IP map has inherited passive TCP/IP header compression:

```
Router> show frame-relay map
Serial 1   (administratively down): ip 131.108.177.177
           dlci 177 (0xB1,0x2C10), static,
           broadcast,
           CISCO
           TCP/IP Header Compression (inherited), passive (inherited)
```

Using an IP Map to Override TCP/IP Header Compression Example

The following example shows the use of a Frame Relay IP map to override the compression set on the interface:

```
interface serial 1
encapsulation frame-relay
ip address 131.108.177.178 255.255.255.0
frame-relay map ip 131.108.177.177 177 broadcast nocompress
frame-relay ip tcp header-compression passive
```

Use of the **show frame-relay map** command will display the resulting compression and encapsulation characteristics; the IP map has not inherited TCP header compression:

```
Serial 1   (administratively down): ip 131.108.177.177
           dlci 177 (0xB1,0x2C10), static,
           broadcast,
           CISCO
```

Disabling TCP/IP Header Compression Examples

The following examples show the use of two different commands to disable TCP/IP header compression.

Disabling Inherited TCP/IP Header Compression Example

In this first example, the initial configuration is the following:

```
interface serial 1
encapsulation frame-relay
ip address 131.108.177.179 255.255.255.0
frame-relay ip tcp header-compression passive
frame-relay map ip 131.108.177.177 177 broadcast
frame-relay map ip 131.108.177.178 178 broadcast tcp header-compression
```

You enter the following commands:

```
serial interface 1
no frame-relay ip tcp header-compression
```

Use of the **show frame-relay map** command will display the resulting compression and encapsulation characteristics:

```
Router> show frame-relay map
Serial 1   (administratively down): ip 131.108.177.177 177
           dlci 177(0xB1, 0x2C10), static,
           broadcast
           CISCO
Serial 1   (administratively down): ip 131.108.177.178 178
           dlci 178(0xB2,0x2C20), static
           broadcast
           CISCO
           TCP/IP Header Compression (enabled)
```

As a result, header compression is disabled for the first map (with DLCI 177), which inherited its header compression characteristics from the interface, but not disabled for the second map (DLCI 178), which is explicitly configured for header compression.

Disabling Explicit TCP/IP Header Compression Example

In this second example, the initial configuration is the same as the previous example, but you enter the following commands:

```
serial interface 1
no frame-relay ip tcp header-compression
frame-relay map ip 131.108.177.178 178 nocompress
```

Use of the **show frame-relay map** command will display the resulting compression and encapsulation characteristics:

```
Router> show frame-relay map
Serial 1   (administratively down): ip 131.108.177.177 177
           dlci 177(0xB1,0x2C10), static,
           broadcast
           CISCO
Serial 1   (administratively down): ip 131.108.177.178 178
           dlci 178(0xB2,0x2C20), static
           broadcast
           CISCO
```

The result of the commands is to disable header compression for the first map (with DLCI 177), which inherited its header compression characteristics from the interface, and also explicitly to disable header compression for the second map (with DLCI 178), which had been explicitly configured for header compression.

Two Routers in Static Mode Example

The following examples illustrate how to configure two routers for static mode.

Configuration for Router 1

```
interface  serial 0
!
ip address  131.108.64.2 255.255.255.0
encapsulation  frame-relay
keepalive 10
frame-relay map ip  131.108.64.1 43
```

Configuration for Router 2

```
interface serial 0
!
ip address 131.108.64.1 255.255.255.0
encapsulation  frame-relay
keepalive 10
frame-relay map ip  131.108.64.2 44
```

Routing DECnet Packets Example

The following example sends all DECnet packets destined for address 56.4 out on DLCI 101. In addition, any DECnet broadcasts for interface serial 1 will be sent on the DLCI.

```
interface serial 1
!
decnet routing 32.6
encapsulation  frame-relay
frame-relay map decnet 56.4 101 broadcast
```

Routing IPX Packets Example

The following example illustrates how to send packets destined for IPX address 200.0000.0c00.7b21 out on DLCI 102:

```
interface ethernet 0
ipx network 2abc
!
interface serial 0
ipx network 200
encapsulation frame-relay
frame-relay map ipx 200.0000.0c00.7b21 102 broadcast
```

Configuration Providing Backward Compatibility Example

The following configuration provides backward compatibility and interoperability. Creating this configuration is possible because of the flexibility provided by separately defining each map entry.

```
encapsulation frame-relay
frame-relay map ip 131.108.123.2 48 broadcast ietf
! interoperability is provided by IETF encapsulation
frame-relay map ip 131.108.123.3 49 broadcast ietf
frame-relay map ip 131.108.123.7 58 broadcast
! this line allows the communication server to connect with a
! device running an older version of software
frame-relay map DECNET 21.7 49 broadcast
```

Configure IETF based on map entries and protocol for more flexibility. Use this method of configuration for backward compatibility and interoperability.

Netbooting over Frame Relay Example

When netbooting over Frame Relay, you cannot netboot via a broadcast. You must netboot from a specific host. Also, a **frame-relay map** command must exist for the host that you will netboot from.

For example, if file gs3-bfx is to be booted from a host with IP address 131.108.126.2, the following commands would need to be in the configuration:

```
boot system gs3-bfx 131.108.126.2

interface Serial 0
encapsulation frame-relay
frame-relay map IP 131.108.126.2 100 broadcast
```

The **frame-relay map** command is used to map an IP address into a DLCI address. In order to netboot over Frame Relay, the address of the machine to netboot from must be given explicitly, and a **frame-relay map** entry must exist for that site. For example:

```
boot system gs3-bfx.83-2.0 131.108.13.111
!
interface Serial 1
ip address 131.108.126.200 255.255.255.0
encapsulation frame-relay
!
frame-relay map IP 131.108.126.111 100 broadcast
```

In this case, 100 is the DLCI of the remote router that can get to host 131.108.126.111.

The remote router must have the following **frame-relay map** entry:

```
frame-relay map IP 131.108.126.200 101 broadcast
```

This entry allows the remote router to return a boot image (from the netboot host) to the router netbooting over Frame Relay. Here, 101 is the DLCI of the router being netbooted.

Transparent Bridging Using Subinterfaces Example

In the following example, Frame Relay DLCIs 42, 64, and 73 are to be used as separate point-to-point links with transparent bridging running over them. The bridging spanning tree algorithm views each PVC as a separate bridge port, and a frame arriving on the PVC can be relayed back out a separate PVC. Be sure that routing is not enabled when configuring transparent bridging using subinterfaces.

```
interface serial 0
bridge-group 1
encapsulation frame-relay
interface serial 0.1
bridge-group 1
frame-relay interface-dlci 42
interface serial 0.2
bridge-group 1
frame-relay interface-dlci 64
interface serial 0.3
bridge-group 1
frame-relay interface-dlci 73
```

PVC Switching Configuration Example

You can configure your router as a dedicated, DCE-only Frame Relay switch. Switching is based on the DLCI. The incoming DLCI is examined, and the outgoing interface and DLCI are determined. Switching takes place when the incoming DLCI in the packet is replaced by the outgoing DLCI, and the packet is sent out the outgoing interface.

In the following example, the router switches two PVCs between interface serial 1 and 2. Frames with DLCI 100 received on serial 1 will be transmitted with DLCI 200 on serial 2 (see Figure 9-2).

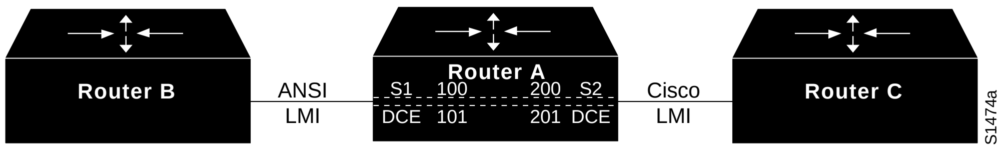

Figure 9-2 PVC Switching Configuration

Configuration for Router A

```
frame-relay switching
!
interface Ethernet0
ip address 131.108.160.58 255.255.255.0
!
interface Serial1
no ip address
encapsulation frame-relay
keepalive 15
frame-relay lmi-type ansi
frame-relay intf-type dce
frame-relay route 100 interface Serial2 200
frame-relay route 101 interface Serial2 201
clockrate 2000000
!
interface Serial2
encapsulation frame-relay
keepalive 15
frame-relay intf-type dce
frame-relay route 200 interface Serial1 100
frame-relay route 201 interface Serial1 101
clockrate 64000
```

Configuring a Pure Frame Relay DCE Example

Using the PVC switching feature, it is possible to build an entire Frame Relay network using our routers. In the following example, Router A and Router C act as Frame Relay switches implementing a two-node network. The standard Network to Network Interface (NNI) signaling protocol is used between Router A and Router C (see Figure 9-3).

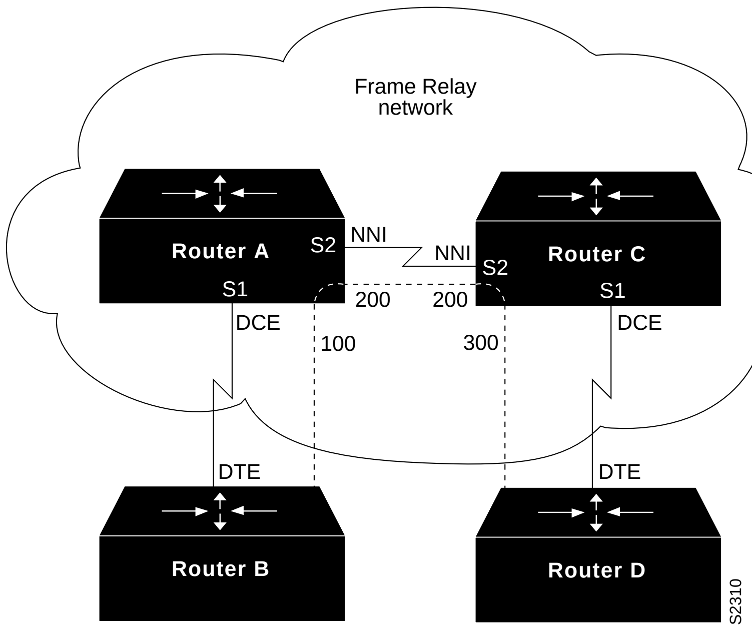

Figure 9-3 Frame Relay DCE Configuration

Configuration for Router A

```
frame-relay switching
!
interface ethernet 0
no ip address
shutdown
!
interface ethernet 1
no ip address
shutdown
!
interface ethernet 2
no ip address
shutdown
!
interface ethernet 3
no ip address
shutdown
!
interface serial 0
ip address 131.108.178.48 255.255.255.0
shutdown
!
interface serial 1
no ip address
encapsulation frame-relay
frame-relay intf-type dce
frame-relay lmi-type ansi
frame-relay route 100 interface serial 2 200
!
interface serial 2
no ip address
encapsulation frame-relay
frame-relay intf-type nni
frame-relay lmi-type q933a
frame-relay route 200 interface serial 1 100
clockrate 2048000
!
interface serial 3
no ip address
shutdown
```

Configuration for Router C

```
frame-relay switching
!
interface ethernet 0
no ip address
shutdown
!
interface ethernet 1
no ip address
shutdown
!
interface ethernet 2
no ip address
shutdown
!
interface ethernet 3
no ip address
shutdown
!
interface serial 0
```

```
ip address 131.108.187.84 255.255.255.0
shutdown
!
interface serial 1
no ip address
encapsulation frame-relay
frame-relay intf-type dce
frame-relay route 300 interface serial 2 200
!
interface serial 2
no ip address
encapsulation frame-relay
frame-relay intf-type nni
frame-relay lmi-type q933a
frame-relay route 200 interface serial 1 300
!
interface serial 3
no ip address
shutdown
```

Hybrid DTE/DCE PVC Switching Example

Routers also can be configured as hybrid DTE/DCE Frame Relay switches (see Figure 9-4).

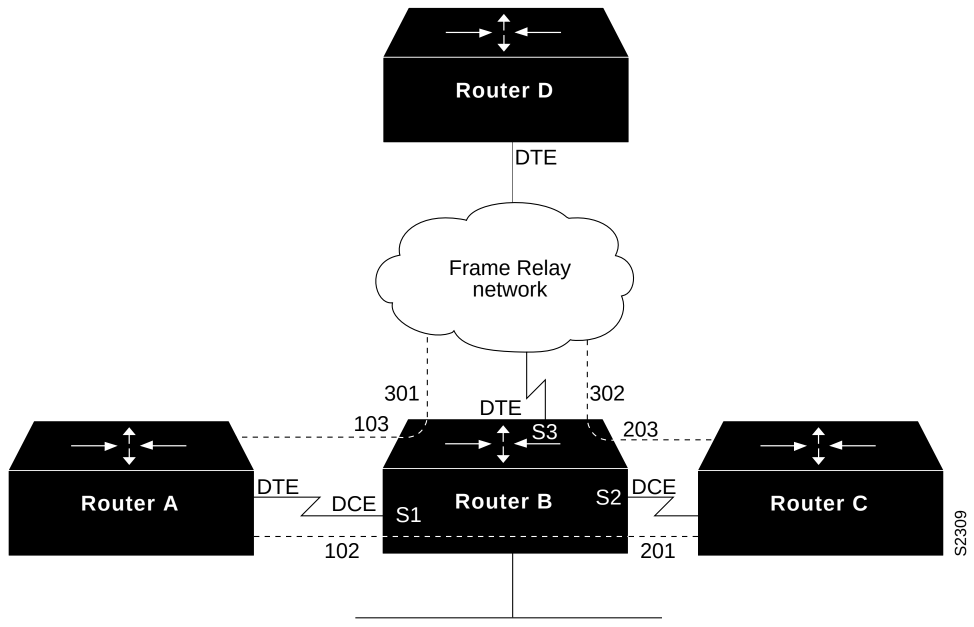

Figure 9-4 Hybrid DTE/DCE PVC Switching

In the following example, Router B acts as a hybrid DTE/DCE Frame Relay switch. It can switch frames between the two DCE ports and between a DCE port and a DTE port. Traffic from the Frame Relay network can also be terminated locally. In the example, three PVCs are defined, as follows:

Serial 1, DLCI 102 to serial 2, DLCI 201	:DCE switching
Serial 1, DLCI 103 to serial 3, DLCI 301	:DCE/DTE switching
Serial 2, DLCI 203 to serial 3, DLCI 302	:DCE/DTE switching

DLCI 400 is also defined for locally terminated traffic.

Configuration for Router B

```
frame-relay switching
!
interface Ethernet0
ip address 131.108.123.231 255.255.255.0
!
interface Ethernet1
ip address 131.108.5.231 255.255.255.0
!
interface Serial0
no ip address
shutdown
!
interface Serial1
no ip address
encapsulation frame-relay
frame-relay intf-type dce
frame-relay route 102 interface serial 2 201
frame-relay route 103 interface serial 3 301
!
interface Serial2
no ip address
encapsulation frame-relay
frame-relay intf-type dce
frame-relay route 201 interface serial 1 102
frame-relay route 203 interface serial 3 302
!
interface Serial3
ip address 131.108.111.231
encapsulation frame-relay
frame-relay lmi-type ansi
frame-relay route 301 interface serial 1 103
frame-relay route 302 interface serial 1 203
frame-relay map ip 131.108.111.4 400 broadcast
```

Switching over an IP Tunnel Example

Switching over an IP tunnel is done by creating a point-to-point tunnel across the internetwork over which PVC switching can take place (see Figure 9-5).

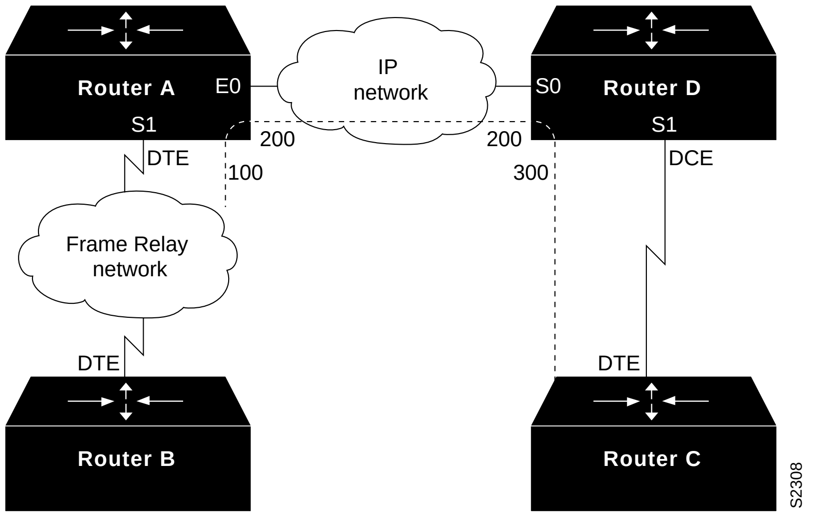

Figure 9-5 Frame Relay Switch over IP Tunnel

The following configurations illustrate how to create the IP network depicted in Figure 9-5.

Configuration for Router A

```
frame-relay switching
!
interface Ethernet0
ip address 108.131.123.231 255.255.255.0
!
interface Ethernet1
ip address 131.108.5.231 255.255.255.0
!
interface Serial0
no ip address
shutdown
!
interface Serial1
ip address 131.108.222.231 255.255.255.0
encapsulation frame-relay
frame-relay map ip 131.108.222.4 400 broadcast
frame-relay route 100 interface Tunnel1 200
!
interface Tunnel1
tunnel source Ethernet0
tunnel destination 150.150.150.123
```

Configuration for Router D

```
frame-relay switching
!
interface Ethernet0
ip address 131.108.231.123 255.255.255.0
!
interface Ethernet1
ip address 131.108.6.123 255.255.255.0
!
interface Serial0
ip address 150.150.150.123 255.255.255.0
encapsulation ppp
!
interface Tunnel1
tunnel source Serial0
tunnel destination 108.131.123.231
!
interface Serial1
ip address 131.108.7.123 255.255.255.0
encapsulation frame-relay
frame-relay intf-type dce
frame-relay route 300 interface Tunnel1 200
```