Wide-Area Networking Configuration Guide: Frame Relay, Cisco IOS XE Release 2

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Wide-Area Networking Overview 1

- Finding Feature Information 1
- Frame Relay 1
  - Frame Relay-ATM Internetworking 3
- Layer 2 Virtual Private Network 4
  - Layer 2 Tunneling Protocol Version 3 4
  - L2VPN Pseudowire Redundancy 4
  - Layer 2 Virtual Private Network Interworking 5
  - Layer 2 Local Switching 5

Configuring Frame Relay 7

- Finding Feature Information 7
- Restrictions for Configuring Frame Relay 7
- Information About Frame Relay 7
  - Frame Relay Hardware Configurations 8
  - Frame Relay Encapsulation 9
  - Dynamic or Static Address Mapping 9
    - Dynamic Address Mapping 9
    - Static Address Mapping 9
  - LMI 9
    - Activating LMI Autosense 10
      - Status Request 10
      - Status Messages 10
      - LMI Autosense 10
      - Configuration Options 10
  - MQC-Based Frame Relay Traffic Shaping 11
    - Traffic-Shaping Map Class for the Interface 11
    - Specifying Map Class with Queueing and Traffic-Shaping Parameters 11
    - Defining Access Lists 11
    - Defining Priority Queue Lists for the Map Class 11

Defining Custom Queue Lists for the Map Class **11**
Understanding Frame Relay Subinterfaces **12**
Subinterface Addressing **13**
Backup Interface for a Subinterface **14**
Disabling or Reenabling Frame Relay Inverse ARP **14**
Frame Relay Fragmentation **14**
End-to-End FRF.12 Fragmentation **15**
Setting the Fragment Size **15**
TCP IP Header Compression **15**
Specifying an Individual IP Map for TCP IP Header Compression **16**
Specifying an Interface for TCP IP Header Compression **16**
Real-Time Header Compression with Frame Relay Encapsulation **16**
Discard Eligibility **17**
DLCI Priority Levels **17**
How to Configure Frame Relay **17**
Enabling Frame Relay Encapsulation on an Interface **18**
Configuring Static Address Mapping **18**
Explicitly Configuring the LMI **18**
Setting the LMI Type **19**
Setting the LMI Keepalive Interval **19**
Setting the LMI Polling and Timer Intervals **19**
Configuring MQC-Based Frame Relay Traffic Shaping **20**
Specifying a Traffic-Shaping Map Class for the Interface **20**
Defining a Map Class with Queueing and Traffic-Shaping Parameters **20**
Customizing Frame Relay for Your Network **21**
Configuring Frame Relay Subinterfaces **21**
Configuring Subinterfaces **21**
Defining Subinterface Addressing on Point-to-Point Subinterfaces **22**
Configuring a Backup Interface for a Subinterface **22**
Disabling or Reenabling Frame Relay Inverse ARP **23**
Configuring Frame Relay Fragmentation **23**
Configuring End-to-End FRF.12 Fragmentation **23**
Verifying the Configuration of End-to-End FRF.12 Fragmentation **23**
Configuring TCP IP Header Compression **24**
Configuring an Individual IP Map for TCP IP Header Compression **24**

Configuring an Interface for TCP IP Header Compression 24
Disabling TCP IP Header Compression 24
Configuring Discard Eligibility 25
Defining a DE List 25
Defining a DE Group 25
Configuring DLCI Priority Levels 25
Monitoring and Maintaining the Frame Relay Connections 26
Configuration Examples for Frame Relay 26
Example IETF Encapsulation 27
Example IETF Encapsulation on the Interface 27
Example IETF Encapsulation on a Per-DLCI Basis 27
Example Static Address Mapping 27
Example Two Routers in Static Mode 27
Example Subinterface 27
Example Basic Subinterface 27
Example Frame Relay Traffic Shaping 28
Example Configuring Class-Based Weighted Fair Queueing 28
Example Configuring Class-Based Weighted Fair Queueing with Fragmentation 28
Example Backward Compatibility 29
Example Booting from a Network Server over Frame Relay 29
Example Frame Relay Fragmentation Configuration 29
Example FRF.12 Fragmentation 29
Example TCP IP Header Compression 30
Example IP Map with Inherited TCP IP Header Compression 30
Example Using an IP Map to Override TCP IP Header Compression 30
Example Disabling Inherited TCP IP Header Compression 31
Example Disabling Explicit TCP IP Header Compression 31
Additional References 32
Feature Information for Configuring Frame Relay 33
Frame Relay Queueing and Fragmentation at the Interface 35
Finding Feature Information 35
Restrictions for Frame Relay Queueing and Fragmentation at the Interface 35
Information About Frame Relay Queueing and Fragmentation at the Interface 36
How Frame Relay Queueing and Fragmentation at the Interface Works 36
Benefits of Frame Relay Queueing and Fragmentation at the Interface 37

How to Configure Frame Relay Queueing and Fragmentation at the Interface 37
Configuring Class Policy for the Priority Queue 38
Configuring Class Policy for the Bandwidth Queues 39
Configuring the Shaping Policy Using the Class-Default Class 40
Configuring Queueing and Fragmentation on the Frame Relay Interface 41
Verifying Frame Relay Queueing and Fragmentation at the Interface 43
Monitoring and Maintaining Frame Relay Queueing and Fragmentation at the Interface 45
Configuration Examples for Frame Relay Queueing and Fragmentation at the Interface 45
Example Frame Relay Queueing Shaping and Fragmentation at the Interface 45
Example Frame Relay Queueing and Fragmentation at the Interface 46
Additional References 46
Feature Information for Frame Relay Queueing and Fragmentation at the Interface 47
Frame Relay MIB Enhancements 49
Finding Feature Information 49
Prerequisites for Frame Relay MIB Enhancements 49
Restrictions for Frame Relay MIB Enhancements 49
Information About Frame Relay MIB Enhancements 50
Feature Overview 50
Benefits 51
How to Configure Frame Relay MIB Enhancements 51
Setting the Load Interval for a PVC 51
Verifying the Load Interval 51
Configuration Examples for Frame Relay MIB Enhancements 52
Example Setting the Load Interval for a PVC 52
Additional References 52
Feature Information for Frame Relay MIB Enhancements 53
Frame Relay PVC Interface Priority Queueing 55
Finding Feature Information 55
Prerequisites for Frame Relay PVC Interface Priority Queueing 55
Restrictions for Frame Relay PVC Interface Priority Queueing 55
Information About Frame Relay PVC Interface Priority Queueing 56
Feature Overview 56
Benefits 57
How to Configure Frame Relay PVC Interface Priority Queueing 57
Configuring PVC Priority in a Map Class 57

Enabling FR PIPQ and Setting Queue Limits 57
Assigning a Map Class to a PVC 58
Verifying FR PIPQ 58
Monitoring and Maintaining FR PIPQ 59
Configuration Examples for Frame Relay PVC Interface Priority Queuing 59
FR PIPQ Configuration Example 59
Additional References 60
Feature Information for Frame Relay PVC Interface Priority Queueing 61
Glossary 62
Frame Relay show Command and debug Command Enhancements 65
Finding Feature Information 65
Information About Frame Relay show Command and debug Command Enhancements 65
Overview of the Frame Relay show Command and debug Command Enhancements 65
Benefits of the Frame Relay Show Command and Debug Command Enhancements 66
Additional References 66
Feature Information for Frame Relay show Command and debug Command Enhancements 67

Wide-Area Networking Overview

Cisco IOS software provides a range of wide-area networking capabilities to fit almost every network environment need. Cisco offers cell relay via the Switched Multimegabit Data Service (SMDS), circuit switching via ISDN, packet switching via Frame Relay, and the benefits of both circuit and packet switching via Asynchronous Transfer Mode (ATM). LAN emulation (LANE) provides connectivity between ATM and other LAN types. The *Cisco IOS Wide-Area Networking Configuration Guide* presents a set of general guidelines for configuring the following software components:

This module gives a high-level description of each technology. For specific configuration information, see the appropriate module.

- Finding Feature Information, page 1
- Frame Relay, page 1
- Layer 2 Virtual Private Network, page 4

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Frame Relay

The Cisco Frame Relay implementation currently supports routing on IP, DECnet, AppleTalk, XNS, Novell IPX, CLNS, Banyan VINES, and transparent bridging.

Although Frame Relay access was originally restricted to leased lines, dialup access is now supported. For more information about dialer profiles or legacy dial-on-demand routing (DDR), see the module Dial-on-Demand Routing Configuration.

To install software on a new router or access server by downloading software from a central server over an interface that supports Frame Relay, see the module Loading and Maintaining System Images.

To configure access between Systems Network Architecture (SNA) devices over a Frame Relay network, see the module Configuring SNA Frame Relay Access Support.

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 ITU-T-adopted Frame Relay signal specification, Q.933 Annex A
- 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 2427, 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 a local data-link connection identifier (DLCI) and a 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.
- Support for both PVCs and SVCs in the same sites and routers.
  - SVCs allow access through a Frame Relay network by setting up a path to the destination endpoints only when the need arises and tearing down the path when it is no longer needed.
- Support for Frame Relay Traffic Shaping beginning with Cisco IOS Release 11.2. Traffic shaping provides the following:
  - Rate enforcement for individual circuits--The peak rate for outbound traffic can be set to the committed information rate (CIR) or some other user-configurable rate.
  - Dynamic traffic throttling on a per-virtual-circuit basis--When backward explicit congestion notification (BECN) packets indicate congestion on the network, the outbound traffic rate is automatically stepped down; when congestion eases, the outbound traffic rate is stepped up again.
  - Enhanced queueing support on a per-virtual circuit basis--Custom queueing, priority queueing, and weighted fair queueing can be configured for individual virtual circuits.
- 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 ARP as described in RFC 1293 for the AppleTalk, Banyan VINES, DECnet, IP, and IPX protocols, and for 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 Media Access Control (MAC)-level address. Routers are configured as a hybrid DTE switch or pure Frame Relay DCE access node in the Frame Relay network.

  Frame Relay switching is used when all traffic arriving on one DLCI can be sent out on another DLCI to the same next-hop address. In such cases, the Cisco IOS software need not examine the frames individually to discover the destination address, and, as a result, the processing load on the router decreases.

The Cisco implementation of Frame Relay switching provides the following functionality:

- Switching over an IP tunnel
- Switching over Network-to-Network Interfaces (NNI) to other Frame Relay switches
- Local serial-to-serial switching
- Switching over ISDN B channels
- Traffic shaping on switched PVCs
- Congestion management on switched PVCs
- Traffic policing on User-Network Interface (UNI) DCE
- FRF.12 fragmentation on switched PVCs

- Support for *subinterfaces* associated with a physical interface. The software groups one or more PVCs under separate subinterfaces, which in turn are located under a single physical interface. See the Configuring Frame Relay module.
- Support for fast-path transparent bridging, as described in RFC 1490, for Frame Relay encapsulated serial and High-Speed Serial Interfaces (HSSIs) on all platforms.
- 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.
- Support for Frame Relay fragmentation. Cisco has developed the following three types of Frame Relay fragmentation:
    - End-to-End FRF.12 Fragmentation

      FRF.12 fragmentation is defined by the FRF.12 Implementation Agreement. This standard was developed to allow long data frames to be fragmented into smaller pieces (fragments) and interleaved with real-time frames. End-to-end FRF.12 fragmentation is recommended for use on PVCs that share links with other PVCs that are transporting voice and on PVCs transporting Voice over IP (VoIP).
    - Frame Relay Fragmentation Using FRF.11 Annex C

      When VoFR (FRF.11) and fragmentation are both configured on a PVC, the Frame Relay fragments are sent in the FRF.11 Annex C format. This fragmentation is used when FRF.11 voice traffic is sent on the PVC, and it uses the FRF.11 Annex C format for data.

      See the module Configuring Voice over Frame Relay in the *Cisco IOS Voice, Video, and Fax Configuration Guide* for configuration tasks and examples for Frame Relay fragmentation using FRF.11 Annex C.
    - Cisco Proprietary Fragmentation

      Cisco proprietary fragmentation is used on data packets on a PVC that is also used for voice traffic.

      See the module Configuring Voice over Frame Relay in the *Cisco IOS Voice, Video, and Fax Configuration Guide* for configuration tasks and examples for Cisco proprietary fragmentation.

- Frame Relay-ATM Internetworking, page 3

Frame Relay-ATM Internetworking

Cisco IOS software supports the Frame Relay Forum implementation agreements for Frame Relay-ATM Interworking. Frame Relay-ATM Interworking enables Frame Relay and ATM networks to exchange data, despite differing network protocols. There are two types of Frame Relay-ATM Interworking:

FRF.5 Frame Relay-ATM Network Interworking

FRF.5 provides network interworking functionality that allows Frame Relay end users to communicate over an intermediate ATM network that supports FRF.5. Multiprotocol encapsulation and other higher-layer procedures are transported transparently, just as they would be over leased lines.

FRF.5 describes network interworking requirements between Frame Relay Bearer Services and Broadband ISDN (BISDN) permanent virtual circuit (PVC) services.

The FRF.5 standard is defined by the Frame Relay Forum Document Number FRF.5: *Frame Relay/ATM PVC Network Interworking Implementation Agreement.* For information about which sections of this implementation agreement are supported by Cisco IOS software, see Frame Relay-ATM Interworking Supported Standards.

FRF.8 Frame Relay-ATM Service Interworking

FRF.8 provides service interworking functionality that allows a Frame Relay end user to communicate with an ATM end user. Traffic is translated by a protocol converter that provides communication among dissimilar Frame Relay and ATM equipment.

FRF.8 describes a one-to-one mapping between a Frame Relay PVC and an ATM PVC.

The FRF.8 standard is defined by the Frame Relay Forum Document Number FRF.8: *Frame Relay/ATM PVC Network Service Interworking Implementation Agreement.* For information about which sections of this implementation agreement are supported by Cisco IOS software, see Frame Relay-ATM Interworking Supported Standards.

Layer 2 Virtual Private Network

L2VPN services are point-to-point. They provide Layer 2 point-to-point connectivity over either an MPLS or a pure IP (L2TPv3) core.

- Layer 2 Tunneling Protocol Version 3, page 4
- L2VPN Pseudowire Redundancy, page 4
- Layer 2 Virtual Private Network Interworking, page 5
- Layer 2 Local Switching, page 5

Layer 2 Tunneling Protocol Version 3

The Layer 2 Tunneling Protocol Version 3 feature expands Cisco's support of Layer 2 VPNs. Layer 2 Tunneling Protocol Version 3 (L2TPv3) is an IETF l2tpext working group draft that provides several enhancements to L2TP to tunnel any Layer 2 payload over L2TP. Specifically, L2TPv3 defines the L2TP protocol for tunneling Layer 2 payloads over an IP core network by using Layer 2 VPNs.

L2VPN Pseudowire Redundancy

L2VPNs can provide pseudowire resiliency through their routing protocols. When connectivity between end-to-end PE routers fails, an alternative path to the directed LDP session and the user data can take over. However, there are some parts of the network where this rerouting mechanism does not protect against interruptions in service. The L2VPN Pseudowire Redundancy feature provides the ability to ensure that the CE2 router in can always maintain network connectivity, even if one or all the failures in the figure occur. The L2VPN Pseudowire Redundancy feature enables you to set up backup pseudowires. You can configure the network with redundant pseudowires (PWs) and redundant network elements.

Layer 2 Virtual Private Network Interworking

Layer 2 transport over MPLS and IP already exists for like-to-like attachment circuits, such as Ethernet-to-Ethernet or PPP-to-PPP. L2VPN Interworking builds on this functionality by allowing disparate attachment circuits to be connected. An interworking function facilitates the translation between the different Layer 2 encapsulations. The L2VPN Interworking feature supports Ethernet, 802.1Q (VLAN), Frame Relay, ATM AAL5, and PPP attachment circuits over MPLS and L2TPv3.

Layer 2 Local Switching

Local switching allows you to switch Layer 2 data between two interfaces of the same type (for example, ATM to ATM, or Frame Relay to Frame Relay) or between interfaces of different types (for example, Frame Relay to ATM) on the same router. The interfaces can be on the same line card or on two different cards. During these kinds of switching, the Layer 2 address is used, not any Layer 3 address. Same-port local switching allows you to switch Layer 2 data between two circuits on the same interface.

Configuring Frame Relay

Frame Relay is a high-performance WAN protocol that operates at the physical and data link layers. The Cisco IOS XE Frame Relay implementation currently supports routing for IPv4, IPv6, and MPLS.

- Finding Feature Information, page 7
- Restrictions for Configuring Frame Relay, page 7
- Information About Frame Relay, page 7
- How to Configure Frame Relay, page 17
- Configuration Examples for Frame Relay, page 26
- Additional References, page 32
- Feature Information for Configuring Frame Relay, page 33

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Configuring Frame Relay

Cisco IOS XE software does not support the following:

- Multipoint permanent virtual circuits (PVCs)
- Switched virtual circuits (SVCs)
- Frame relay switching
- 4-byte extended addresses
- End-to-end keepalives
- FRF.9 payload compression
- Legacy frame-relay traffic shaping (Cisco IOS XE software supports only policy map-based MQC.)

Information About Frame Relay

- Frame Relay Hardware Configurations, page 8
- Frame Relay Encapsulation, page 9
- Dynamic or Static Address Mapping, page 9
- LMI, page 9
- MQC-Based Frame Relay Traffic Shaping, page 11
- Understanding Frame Relay Subinterfaces, page 12
- Disabling or Reenabling Frame Relay Inverse ARP, page 14
- Frame Relay Fragmentation, page 14
- TCP IP Header Compression, page 15
- Real-Time Header Compression with Frame Relay Encapsulation, page 16
- Discard Eligibility, page 17
- DLCI Priority Levels, page 17

Frame Relay Hardware Configurations

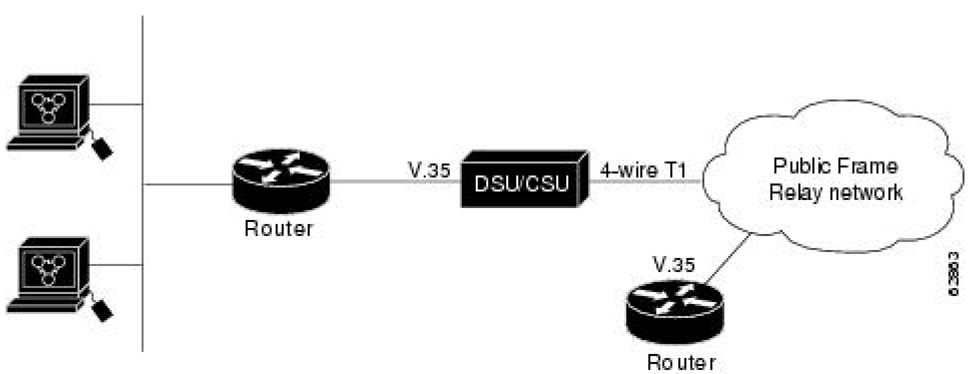
You can create Frame Relay connections using one of the following hardware configurations:

- Routers and access servers connected directly to the Frame Relay switch
- Routers and access servers connected directly to a channel service unit/digital service unit (CSU/DSU), which then connects to a remote Frame Relay switch

Note Routers can connect to Frame Relay networks either by direct connection to a Frame Relay switch, through a direct connection to a POS interface or a T1/T3 interface, or through CSU/DSUs. However, a single router interface configured for Frame Relay can be configured for only one of these methods.

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. The figure below illustrates the connections among the components.

Figure 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.

Frame Relay Encapsulation

Frame Relay supports encapsulation of all supported protocols in conformance with RFC 1490, allowing interoperability among multiple vendors. Use the IETF form of Frame Relay encapsulation if your router or access server is connected to another vendor’s equipment across a Frame Relay network. IETF encapsulation is supported either at the interface level or on a per-VC basis.

Shut down the interface prior to changing encapsulation types. Although shutting down the interface is not required, it ensures that the interface is reset for the new encapsulation.

Dynamic or Static Address Mapping

- Dynamic Address Mapping, page 9
- Static Address Mapping, page 9

Dynamic Address Mapping

Dynamic address mapping uses Frame Relay Inverse ARP to request the next-hop protocol address for a specific connection, given its known DLCI. Responses to Inverse ARP requests are entered in an address-to-DLCI mapping table on the router or access server; the table is then used to supply the next-hop protocol address or the DLCI for outgoing traffic.

Inverse ARP is enabled by default for all protocols it supports, but can be disabled for specific protocol-DLCI pairs. As a result, you can use dynamic mapping for some protocols and static mapping for other protocols on the same DLCI. You can explicitly disable Inverse ARP for a protocol-DLCI pair if you know that the protocol is not supported on the other end of the connection. For more information, see the Disabling or Reenabling Frame Relay Inverse ARP section.

Note Because Inverse ARP is enabled by default, no additional command is required to configure dynamic mapping on an interface and packets are not sent out for protocols that are not enabled on the interface.

Static Address Mapping

A static map links a specified next-hop protocol address to a specified DLCI. Static mapping removes the need for Inverse ARP requests; when you supply a static map, Inverse ARP is automatically disabled for the specified protocol on the specified DLCI. You must use static mapping if the router at the other end either does not support Inverse ARP at all or does not support Inverse ARP for a specific protocol that you want to use over Frame Relay.

You can simplify the configuration for the Open Shortest Path First (OSPF) protocol by adding the optional **broadcast** keyword when doing this task. Refer to the **frame-relay map** command description in the *Cisco IOS Wide-Area Networking Command Reference* and the examples at the end of this chapter for more information about using the **broadcast**keyword.

LMI

The Cisco IOS XE software supports Local Management Interface (LMI) autosense, which enables the interface to determine the LMI type supported by the switch. Support for LMI autosense means that you are no longer required to configure the LMI explicitly.

LMI autosense is active in the following situations:

- The router is powered up or the interface changes state to up.
- The line protocol is down but the line is up.
- The interface is a Frame Relay DTE.
- The LMI type is not explicitly configured.

- Activating LMI Autosense, page 10

Activating LMI Autosense

- Status Request, page 10
- Status Messages, page 10
- LMI Autosense, page 10
- Configuration Options, page 10

Status Request

When LMI autosense is active, it sends out a full status request, in all three LMI types, to the switch. The order is ANSI, ITU, cisco, but it is done in rapid succession. Cisco IOS XE software provides the ability to listen in on both DLCI 1023 (cisco LMI) and DLCI 0 (ANSI and ITU) simultaneously.

Status Messages

One or more of the status requests will elicit a reply (status message) from the switch. The router will decode the format of the reply and configure itself automatically. If more than one reply is received, the router will configure itself with the type of the last received reply. This is to accommodate intelligent switches that can handle multiple formats simultaneously.

LMI Autosense

If LMI autosense is unsuccessful, an intelligent retry scheme is built in. Every N391 interval (default is 60 seconds, which is 6 keep exchanges at 10 seconds each), LMI autosense will attempt to ascertain the LMI type. For more information about N391, see the **frame-relay lmi-n391dte** command in the chapter "Frame Relay Commands " in the *Cisco IOS Wide-Area Networking Command Reference* .

The only visible indication to the user that LMI autosense is under way is that **debug frame lmi** is turned on. At every N391 interval, the user will now see three rapid status inquiries coming out of the serial interface: one in ANSI, one in ITU, and one in cisco LMI-type.

Configuration Options

No configuration options are provided; LMI autosense is transparent to the user. You can turn off LMI autosense by explicitly configuring an LMI type. The LMI type must be written into NVRAM so that next time the router powers up, LMI autosense will be inactive. At the end of autoinstall, a **frame-relay lmi-type** *xxx* statement is included within the interface configuration. This configuration is not automatically written to NVRAM; you must explicitly write the configuration to NVRAM by using the **copy system:running-config** or **copy nvram:startup-config** command.

MQC-Based Frame Relay Traffic Shaping

Legacy frame-relay traffic shaping is not supported. Cisco IOS XE software only supports policy map based MQC.

- Traffic-Shaping Map Class for the Interface, page 11
- Specifying Map Class with Queueing and Traffic-Shaping Parameters, page 11
- Defining Access Lists, page 11
- Defining Priority Queue Lists for the Map Class, page 11
- Defining Custom Queue Lists for the Map Class, page 11

Traffic-Shaping Map Class for the Interface

If you specify a Frame Relay map class for a main interface, all the VCs on its subinterfaces inherit all the traffic-shaping parameters defined for the class. You can override the default for a specific DLCI on a specific subinterface by using the **class** VC configuration command to assign the DLCI explicitly to a different class. See the section Configuring Frame Relay Subinterfaces, page 21 for information about setting up subinterfaces.

Specifying Map Class with Queueing and Traffic-Shaping Parameters

When defining a map class for Frame Relay, you can specify the average and peak rates (in bits per second) allowed on VCs associated with the map class. You can also specify *either* a custom queue list *or* a priority queue group to use on VCs associated with the map class.

Defining Access Lists

You can specify access lists and associate them with the custom queue list defined for any map class. The list number specified in the access list and the custom queue list tie them together. See the appropriate protocol chapters for information about defining access lists for the protocols you want to transmit on the Frame Relay network.

Defining Priority Queue Lists for the Map Class

You can define a priority list for a protocol and you can also define a default priority list. The number used for a specific priority list ties the list to the Frame Relay priority group defined for a specified map class. For example, if you enter the **frame relay priority-group 2** command for the map class "fast_vcs" and then you enter the **priority-list 2 protocol decnet high** command, that priority list is used for the "fast_vcs" map class. The average and peak traffic rates defined for the "fast_vcs" map class are used for DECnet traffic.

Defining Custom Queue Lists for the Map Class

You can define a queue list for a protocol and a default queue list. You can also specify the maximum number of bytes to be transmitted in any cycle. The number used for a specific queue list ties the list to the Frame Relay custom queue list defined for a specified map class.

For example, if you enter the **frame relay custom-queue-list 1** command for the map class "slow_vcs" and then you enter the **queue-list 1 protocol ip list 100** command, that queue list is used for the "slow_vcs" map class; **access-list 100** definition is also used for that map class and queue. The average and peak traffic rates defined for the "slow_vcs" map class are used for IP traffic that meets the **access list 100** criteria.

Understanding Frame Relay Subinterfaces

Frame Relay subinterfaces provide a mechanism for supporting partially meshed Frame Relay networks. Most protocols assume transitivity on a logical network; that is, if station A can talk to station B, and station B can talk to station C, then station A should be able to talk to station C directly. Transitivity is true on LANs, but not on Frame Relay networks unless A is directly connected to C.

Additionally, certain protocols such as AppleTalk and transparent bridging cannot be supported on partially meshed networks because they require *split horizon* . Split horizon is a routing technique in which a packet received on an interface cannot be sent from the same interface even if received and transmitted on different VCs.

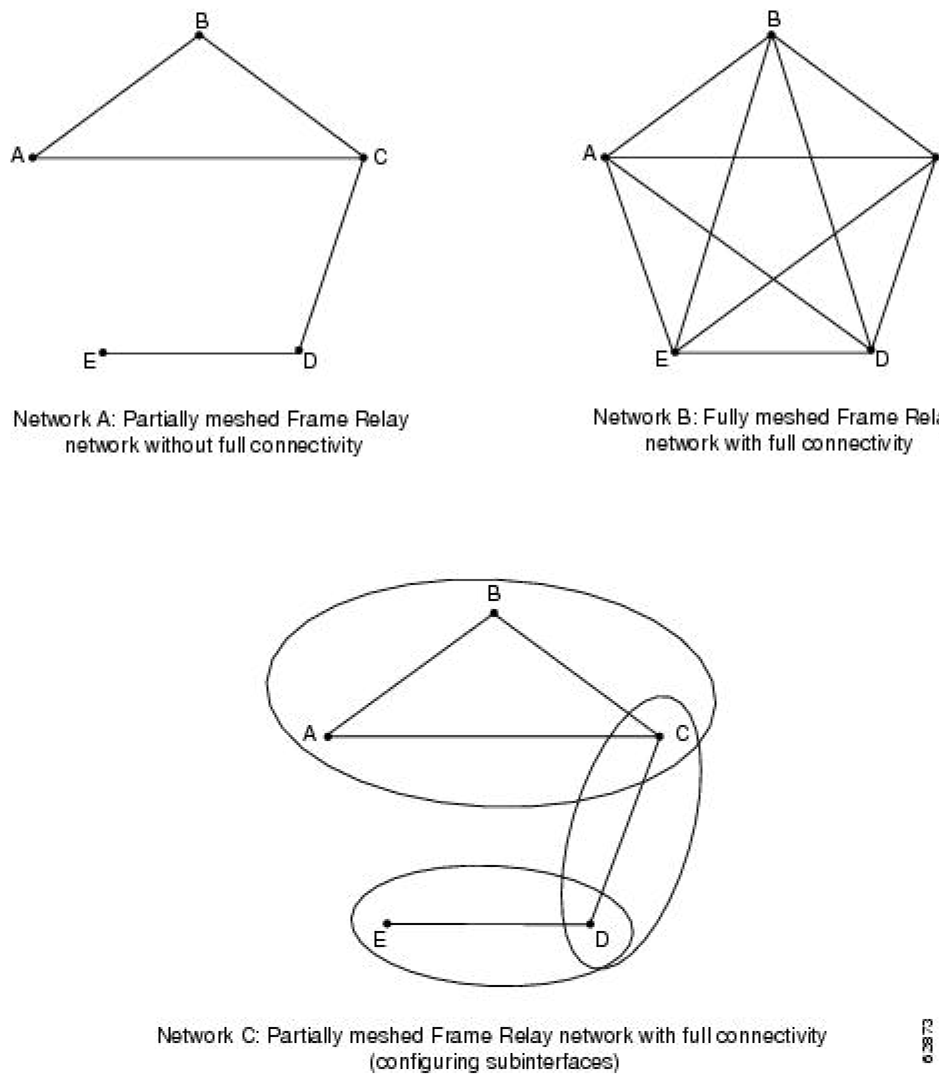
Configuring Frame Relay subinterfaces ensures that a single physical interface is treated as multiple virtual interfaces. This treatment allows you to overcome split horizon rules. Packets received on one virtual interface can be forwarded to another virtual interface even if they are configured on the same physical interface.

Subinterfaces address the limitations of Frame Relay networks by providing a way to subdivide a partially meshed Frame Relay network into a number of smaller, fully meshed (or point-to-point) subnetworks. Each subnetwork is assigned its own network number and appears to the protocols as if it were reachable through a separate interface. (Note that point-to-point subinterfaces can be unnumbered for use with IP, reducing the addressing burden that might otherwise result.)

Note Cisco IOS XE software supports configuration of point-to-point subinterfaces.

The figure below shows a five-node Frame Relay network that is partially meshed (network A). If the entire network is viewed as a single subnetwork (with a single network number assigned), most protocols assume that node A can transmit a packet directly to node E, when in fact it must be relayed through nodes C and D. This network can be made to work with certain protocols (for example, IP), but will not work at all with other protocols (for example, AppleTalk) because nodes C and D will not relay the packet out the same interface on which it was received. One way to make this network work fully is to create a fully meshed network (network B), but doing so requires a large number of PVCs, which may not be economically feasible.

Figure 2 ***Using Subinterfaces to Provide Full Connectivity on a Partially Meshed Frame Relay Network***



Using subinterfaces, you can subdivide the Frame Relay network into three smaller subnetworks (network C) with separate network numbers. Nodes A, B, and C are connected to a fully meshed network, and nodes C and D, as well as nodes D and E, are connected via point-to-point networks. In this configuration, nodes C and D can access two subinterfaces and can therefore forward packets without violating split horizon rules. If transparent bridging is being used, each subinterface is viewed as a separate bridge port.

- Subinterface Addressing, page 13
- Backup Interface for a Subinterface, page 14

Subinterface Addressing

For point-to-point subinterfaces, the destination is presumed to be known and is identified or implied in the **frame-relay interface-dlci** command.

Note The **frame-relay interface-dlci** command is typically used on subinterfaces; however, it can also be applied to main interfaces. The command is used to enable routing protocols on main interfaces that are configured to use Inverse ARP. This command is also helpful for assigning a specific class to a single PVC on a multipoint subinterface.

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 or access server. Instead, you can simply avoid using that subinterface number and use a different subinterface number.

Backup Interface for a Subinterface

Both point-to-point and multipoint Frame Relay subinterfaces can be configured with a backup interface. This approach allows individual PVCs to be backed up in case of failure rather than depending on the entire Frame Relay connection to fail before the backup takes over. You can configure a subinterface for backup on failure only, not for backup based on loading of the line.

If the main interface has a backup interface, it will have precedence over the subinterface's backup interface in the case of complete loss of connectivity with the Frame Relay network. As a result, a subinterface backup is activated only if the main interface is up, or if the interface is down and does not have a backup interface defined. If a subinterface fails while its backup interface is in use, and the main interface goes down, the backup subinterface remains connected.

Disabling or Reenabling Frame Relay Inverse ARP

Frame Relay Inverse ARP is a method of building dynamic address mappings in Frame Relay networks running DECnet, IP, and Novell IPX. Inverse ARP allows the router or access server to discover the protocol address of a device associated with the VC.

Inverse ARP creates dynamic address mappings, as contrasted with the **frame-relay map** command, which defines static mappings between a specific protocol address and a specific DLCI (see the section Configuring Static Address Mapping, page 18 for more information).

Inverse ARP is enabled by default but can be disabled explicitly for a given protocol and DLCI pair. Disable or reenable Inverse ARP under the following conditions:

- Disable Inverse ARP for a selected protocol and DLCI pair when you know that the protocol is not supported at the other end of the connection.
- Reenable Inverse ARP for a protocol and DLCI pair if conditions or equipment change and the protocol is then supported at the other end of the connection.

Note If you change from a point-to-point subinterface to a multipoint subinterface, change the subinterface number. Frame Relay Inverse ARP will be on by default, and no further action is required.

You do not need to enable or disable Inverse ARP if you have a point-to-point interface, because there is only a single destination and discovery is not required.

Frame Relay Fragmentation

- End-to-End FRF.12 Fragmentation, page 15

End-to-End FRF.12 Fragmentation

The purpose of end-to-end FRF.12 fragmentation is to support real-time and non-real-time data packets on lower-speed links without causing excessive delay to the real-time data. FRF.12 fragmentation is defined by the FRF.12 Implementation Agreement. This standard was developed to allow long data frames to be fragmented into smaller pieces (fragments) and interleaved with real-time frames. In this way, real-time and non-real-time data frames can be carried together on lower-speed links without causing excessive delay to the real-time traffic.

End-to-end FRF.12 fragmentation is recommended for use on permanent virtual circuits (PVCs) that share links with other PVCs that are transporting voice and on PVCs transporting Voice over IP (VoIP). Although VoIP packets should not be fragmented, they can be interleaved with fragmented packets.

FRF.12 is configured on a per-PVC basis using a Frame Relay map class. The map class can be applied to one or many PVCs. Frame Relay traffic shaping must be enabled on the interface in order for fragmentation to work.

Note When Frame Relay fragmentation is configured, WFQ or LLQ is mandatory. If a map class is configured for Frame Relay fragmentation and the queueing type on that map class is not WFQ or LLQ, the configured queueing type is automatically overridden by WFQ with the default values. To configure LLQ for Frame Relay, refer to the *Cisco IOS XE Quality of Service Solutions Configuration Guide* .

- Setting the Fragment Size, page 15

Setting the Fragment Size

Set the fragment size so that voice packets are not fragmented and do not experience a serialization delay greater than 20 ms.

To set the fragment size, the link speed must be taken into account. The fragment size should be larger than the voice packets, but small enough to minimize latency on the voice packets. Turn on fragmentation for low speed links (less than 768 kb/s).

Set the fragment size based on the lowest port speed between the routers. For example, if there is a hub and spoke Frame Relay topology where the hub has a T1 speed and the remote routers have 64 kb/s port speeds, the fragment size needs to be set for the 64 kb/s speed on both routers. Any other PVCs that share the same physical interface need to configure the fragmentation to the size used by the voice PVC.

If the lowest link speed in the path is 64 kb/s, the recommended fragment size (for 10 ms serialization delay) is 80 bytes. If the lowest link speed is 128 kb/s, the recommended fragment size is 160 bytes.

For more information, refer to the " Fragmentation (FRF.12)" section in the VoIP over Frame Relay with Quality of Service (Fragmentation, Traffic Shaping, LLQ / IP RTP Priority) document.

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. Reconstructing a smaller header that identifies the connection and indicates the fields that have changed and the amount of change reduces the number of bytes 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 queueing changes the order in which packets are transmitted, enabling priority queueing on the interface is not recommended.

Note If you configure an interface with Cisco-proprietary 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.

- Specifying an Individual IP Map for TCP IP Header Compression, page 16
- Specifying an Interface for TCP IP Header Compression, page 16

Specifying an Individual IP Map for TCP IP Header Compression

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

TCP/IP header compression requires Cisco-proprietary 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-proprietary 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 a packet had a compressed TCP/IP header when it was received.

Specifying 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.

Note If an interface configured with Cisco-proprietary 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 Configuring an Individual IP Map for TCP IP Header Compression section.

Real-Time Header Compression with Frame Relay Encapsulation

Real-time Transport Protocol (RTP) is a protocol used for carrying packetized audio and video traffic over an IP network, providing end-to-end network transport functions intended for these real-time traffic applications and multicast or unicast network services. RTP is described in RFC 1889. RTP is not intended for data traffic, which uses TCP or UDP.

For configuration tasks for and examples of RTP header compression using Frame Relay encapsulation, see the *Cisco IOS XE IP Multicast Configuration Guide* .

The commands for configuring this feature appear in the *Cisco IOS IP Multicast Command Reference*.

Discard Eligibility

Some Frame Relay packets can be set with low priority or low time sensitivity. These 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.

Discard eligibility requires the Frame Relay network to be able to interpret the DE bit. Some networks take no action when the DE bit is set, and others use the DE bit to determine which packets to discard. The best 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 create 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.

You can create DE lists based on the protocol or the interface, and on characteristics such as fragmentation of the packet, a specific TCP or User Datagram Protocol (UDP) port, an access list number, or a packet size.

DLCI Priority Levels

DLCI priority levels allow you to separate different types of traffic and provides a traffic management tool for congestion problems caused by following situations:

- Mixing batch and interactive traffic over the same DLCI
- Queueing traffic from sites with high-speed access at destination sites with lower-speed access

Before you configure the DLCI priority levels, you must:

- Define a global priority list.
- Enable Frame Relay encapsulation.
- Define dynamic or static address mapping.
- Make sure that you define each of the DLCIs to which you intend to apply levels. You can associate priority-level DLCIs with subinterfaces.
- Configure the LMI.

Note DLCI priority levels provide a way to define multiple parallel DLCIs for different types of traffic. DLCI priority levels do not assign priority queues within the router or access server. In fact, they are independent of the device’s priority queues. However, if you enable queueing and use the same DLCIs for queueing, then high-priority DLCIs can be put into high-priority queues.

How to Configure Frame Relay

- Enabling Frame Relay Encapsulation on an Interface, page 18
- Configuring Static Address Mapping, page 18
- Explicitly Configuring the LMI, page 18

- Configuring MQC-Based Frame Relay Traffic Shaping, page 20
- Customizing Frame Relay for Your Network, page 21
- Monitoring and Maintaining the Frame Relay Connections, page 26

Enabling Frame Relay Encapsulation on an Interface

Note Frame Relay encapsulation is a prerequisite for any Frame Relay commands on an interface.

To enable Frame Relay encapsulation on the interface level, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **interface** *typenumber*
2. Router(config-if)# **encapsulation frame-relay**[**ietf**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface** *typenumber*	Specifies the interface, and enters interface configuration mode.
Step 2	Router(config-if)# **encapsulation frame-relay**[**ietf**]	Enables and specifies the Frame Relay encapsulation method.

Configuring Static Address Mapping

To establish static mapping according to your network needs, use the following command in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **frame-relay map** *protocol protocol-address dlci* [**broadcast**] [**ietf**] [**cisco**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay map** *protocol protocol-address dlci* [**broadcast**] [**ietf**] [**cisco**]	Enables and specifies the Frame Relay encapsulation method.

Explicitly Configuring the LMI

- Setting the LMI Type, page 19
- Setting the LMI Keepalive Interval, page 19
- Setting the LMI Polling and Timer Intervals, page 19

Setting the LMI Type

If the router or access server is attached to a public data network (PDN), the LMI type must match the type used on the public network. Otherwise, the LMI type can be set to suit the needs of your private Frame Relay network. You can set one of the following three types of LMIs on Cisco devices: ANSI T1.617 Annex D, Cisco, and ITU-T Q.933 Annex A. To do so, use the following commands beginning in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **frame-relay lmi-type** {**ansi** | **cisco** | **q933a**}
2. Router# **copy nvram:startup-config** *destination*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay lmi-type** {**ansi** | **cisco** | **q933a**}	Sets the LMI type.
Step 2	Router# **copy nvram:startup-config** *destination*	Writes the LMI type to NVRAM.

Setting the LMI Keepalive Interval

A keepalive interval must be set to configure the LMI. By default, this interval is 10 seconds and, according to the LMI protocol, must be less than the corresponding interval on the switch. To set the keepalive interval, use the following command in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **keepalive** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **keepalive** *number*	Writes the LMI type to NVRAM. **Note** To disable keepalives on networks that do not utilize LMI, use the **no keepalive** command.

Setting the LMI Polling and Timer Intervals

You can set various optional counters, intervals, and thresholds to fine-tune the operation of your LMI DTE and DCE devices. Set these attributes by using one or more of the following commands in interface configuration mode:

Command	Purpose
frame-relay lmi-n392dce *threshold*	Sets the DCE and Network-to-Network Interface (NNI) error threshold.

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

Configuring MQC-Based Frame Relay Traffic Shaping

- Specifying a Traffic-Shaping Map Class for the Interface, page 20
- Defining a Map Class with Queueing and Traffic-Shaping Parameters, page 20

Specifying a Traffic-Shaping Map Class for the Interface

To specify a map class for the specified interface, use the following command beginning in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **frame-relay class** *map-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay class** *map-class-name*	Specifies a Frame Relay map class for the interface.

Defining a Map Class with Queueing and Traffic-Shaping Parameters

To define a map class, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **map-class frame-relay** *map-class-name*
2. Router(config-map-class)# **frame-relay traffic-rate** *average* [*peak*]
3. Router(config-map-class)# **frame-relay custom-queue-list** *list-number*
4. Router(config-map-class)# **frame-relay priority-group** *list-number*
5. Router(config-map-class)# **frame-relay adaptive-shaping** {**becn** | **foresight**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **map-class frame-relay** *map-class-name*	Specifies a map class to define.
Step 2	Router(config-map-class)# **frame-relay traffic-rate** *average* [*peak*]	Defines the traffic rate for the map class.
Step 3	Router(config-map-class)# **frame-relay custom-queue-list** *list-number*	Specifies a custom queue list.
Step 4	Router(config-map-class)# **frame-relay priority-group** *list-number*	Specifies a priority queue list.
Step 5	Router(config-map-class)# **frame-relay adaptive-shaping** {**becn** | **foresight**}	Selects BECN or ForeSight as congestion backward-notification mechanism to which traffic shaping adapts.

Customizing Frame Relay for Your Network

- Configuring Frame Relay Subinterfaces, page 21
- Disabling or Reenabling Frame Relay Inverse ARP, page 23
- Configuring Frame Relay Fragmentation, page 23
- Configuring TCP IP Header Compression, page 24
- Configuring Discard Eligibility, page 25
- Configuring DLCI Priority Levels, page 25

Configuring Frame Relay Subinterfaces

- Configuring Subinterfaces, page 21
- Defining Subinterface Addressing on Point-to-Point Subinterfaces, page 22
- Configuring a Backup Interface for a Subinterface, page 22

Configuring Subinterfaces

Note Multipoint DLCI configurations are currently not supported. Cisco IOS XE software supports point-to-point connections.

To configure subinterfaces on a Frame Relay network, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **interface type** *number* **.** *subinterface-number* {**multipoint** | **point-to-point**}
2. Router(config-subif)# **encapsulation frame-relay**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface type** *number* **.** *subinterface-number* {**multipoint** | **point-to-point**}	Creates a point-to-point or multipoint subinterface. • Cisco IOS XE software only supports point-to-point subinterfaces.
Step 2	Router(config-subif)# **encapsulation frame-relay**	Configures Frame Relay encapsulation on the serial interface.

Defining Subinterface Addressing on Point-to-Point Subinterfaces

If you specified a point-to-point subinterface in the preceding procedure, use the following command in subinterface configuration mode:

SUMMARY STEPS

1. Router(config-subif)# **frame-relay interface-dlci** *dlci*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-subif)# **frame-relay interface-dlci** *dlci*	Associates the selected point-to-point subinterface with a DLCI.

Configuring a Backup Interface for a Subinterface

To configure a backup interface for a Frame Relay subinterface, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **interface type** *number*
2. Router(config-if)# **encapsulation frame-relay**
3. Router(config)# **interface type** *number* **.** *subinterface-number* **point-to-point**
4. Router(config-subif)# **frame-relay interface-dlci** *dlci*
5. Router(config-subif)# **backup interface type** *number*
6. Router(config-subif)# **backup delay** *enable-delay disable-delay*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface type** *number*	Specifies the interface.
Step 2	Router(config-if)# **encapsulation frame-relay**	Configures Frame Relay encapsulation.
Step 3	Router(config)# **interface type** *number* **.** *subinterface-number* **point-to-point**	Configures the subinterface.

	Command or Action	Purpose
Step 4	Router(config-subif)# **frame-relay interface-dlci** *dlci*	Specifies DLCI for the subinterface.
Step 5	Router(config-subif)# **backup interface type** *number*	Configures backup interface for the subinterface.
Step 6	Router(config-subif)# **backup delay** *enable-delay disable-delay*	Specifies backup enable and disable delay.

Disabling or Reenabling Frame Relay Inverse ARP

To select or disable Inverse ARP, use one of the following commands in interface configuration mode:

Command	Purpose
frame-relay inverse-arp *protocol dlci*	Enables Frame Relay Inverse ARP for a specific protocol and DLCI pair, only if it was previously disabled.
no frame relay inverse-arp *protocol dlci*	Disables Frame Relay Inverse ARP for a specific protocol and DLCI pair.

Configuring Frame Relay Fragmentation

- Configuring End-to-End FRF.12 Fragmentation, page 23
- Verifying the Configuration of End-to-End FRF.12 Fragmentation, page 23

Configuring End-to-End FRF.12 Fragmentation

To configure FRF.12 fragmentation in a Frame Relay map class, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **map-class frame-relay** *map-class-name*
2. Router(config-map-class)# **frame-relay fragment** *fragment_size*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **map-class frame-relay** *map-class-name*	Specifies a map class to define QoS values for a Frame Relay SVC or PVC. The map class can be applied to one or many PVCs.
Step 2	Router(config-map-class)# **frame-relay fragment** *fragment_size*	Configures Frame Relay fragmentation for the map class. The *fragment_size* argument defines the payload size of a fragment; it excludes the Frame Relay headers and any Frame Relay fragmentation header. The valid range is from 16 to 1600 bytes, and the default is 53.

Verifying the Configuration of End-to-End FRF.12 Fragmentation

To verify FRF.12 fragmentation, use one or more of the following EXEC commands:

Command	Purpose
show frame-relay fragment [**interface** *interface*] [*dlci*]	Displays Frame Relay fragmentation information.
show frame-relay pvc [**interface** *interface*] [*dlci*]	Displays statistics about PVCs for Frame Relay interfaces.

Configuring TCP IP Header Compression

- Configuring an Individual IP Map for TCP IP Header Compression, page 24
- Configuring an Interface for TCP IP Header Compression, page 24
- Disabling TCP IP Header Compression, page 24

Configuring an Individual IP Map for TCP IP Header Compression

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

Command	Purpose
frame-relay map ip *ip-address dlci* [**broadcast**] **tcp header-compression** [**active** | **passive**] [**connections** *number*]	Configures an IP map to use TCP/IP header compression. Cisco-proprietary encapsulation is enabled by default.

Configuring an Interface for TCP IP Header Compression

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

SUMMARY STEPS

1. Router(config-if)# **encapsulation frame-relay**
2. Router(config-if)# **frame-relay ip tcp header-compression** [**passive**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **encapsulation frame-relay**	Configures Cisco-proprietary encapsulation on the interface.
Step 2	Router(config-if)# **frame-relay ip tcp header-compression** [**passive**]	Enables TCP/IP header compression.

Disabling 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, use one of the following commands in interface configuration mode:

Command	Purpose
no frame-relay ip tcp header-compression	Disables TCP/IP header compression on all Frame Relay IP maps that are not explicitly configured for TCP header compression.
frame-relay map ip *ip-address dlci* **nocompress**	Disables RTP and TCP/IP header compression on a specified Frame Relay IP map.

Configuring Discard Eligibility

- Defining a DE List, page 25
- Defining a DE Group, page 25

Defining a DE List

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

SUMMARY STEPS

1. Router(config)# **frame-relay de-list** *list-number* {**protocol** *protocol* | **interface** *type number*} *characteristic*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **frame-relay de-list** *list-number* {**protocol** *protocol*	**interface** *type number*} *characteristic*	Defines a DE list.

Defining a DE Group

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

Command	Purpose
frame-relay de-group *group-number* `dlci`	Defines a DE group.

Configuring DLCI Priority Levels

To configure DLCI priority levels, use the following command in interface configuration mode:

Command	Purpose
frame-relay priority-dlci-group *group-number* *high-dlci* *medium-dlci* *normal-dlci* *low-dlci*	Enables multiple parallel DLCIs for different Frame Relay traffic types; associates and sets level of specified DLCIs with same group. **Note** If you do not explicitly specify a DLCI for each of the priority levels, the last DLCI specified in the command line is used as the value of the remaining arguments. At a minimum, you must configure the high-priority and the medium-priority DLCIs.

Monitoring and Maintaining the Frame Relay Connections

To monitor Frame Relay connections, use any of the following commands in EXEC mode:

Command	Purpose
clear frame-relay-inarp	Clears dynamically created Frame Relay maps, which are created by the use of Inverse ARP.
show interfaces serial *type number*	Displays information about Frame Relay DLCIs and the LMI.
show frame-relay lmi [*type number*]	Displays LMI statistics.
show frame-relay map	Displays the current Frame Relay map entries.
show frame-relay pvc [*type number* [*dlci*]]	Displays PVC statistics.
show frame-relay route	Displays configured static routes.
show frame-relay traffic	Displays Frame Relay traffic statistics.
show frame-relay lapf	Displays information about the status of LAPF.
show frame-relay svc maplist	Displays all the SVCs under a specified map list.

Configuration Examples for Frame Relay

- Example IETF Encapsulation, page 27
- Example Static Address Mapping, page 27
- Example Subinterface, page 27
- Example Frame Relay Traffic Shaping, page 28
- Example Backward Compatibility, page 29
- Example Booting from a Network Server over Frame Relay, page 29
- Example Frame Relay Fragmentation Configuration, page 29
- Example TCP IP Header Compression, page 30

Example IETF Encapsulation

- Example IETF Encapsulation on the Interface, page 27
- Example IETF Encapsulation on a Per-DLCI Basis, page 27

Example IETF Encapsulation on the Interface

The following example sets IETF encapsulation at the interface level. 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
```

Example IETF Encapsulation on a Per-DLCI Basis

The following example configures IETF encapsulation 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
```

Example Static Address Mapping

- Example Two Routers in Static Mode, page 27

Example Two Routers in Static Mode

The following example shows how to configure two routers for static mode:

Configuration for Router 1

```
interface serial0
 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 serial1
 ip address 131.108.64.1 255.255.255.0
 encapsulation frame-relay
 keepalive 10
 frame-relay map ip 131.108.64.2 43
```

Example Subinterface

- Example Basic Subinterface, page 27

Example Basic Subinterface

In the following example, subinterface 1 is configured as a point-to-point subnet and subinterface 2 is configured as a multipoint subnet.

```
interface serial 0
 encapsulation frame-relay
interface serial 0.1 point-to-point
 ip address 10.0.1.1 255.255.255.0
 frame-relay interface-dlci 42
!
interface serial 0.2 multipoint
 ip address 10.0.2.1 255.255.255.0
 frame-relay map ip 10.0.2.2 18
```

Example Frame Relay Traffic Shaping

- Example Configuring Class-Based Weighted Fair Queueing, page 28
- Example Configuring Class-Based Weighted Fair Queueing with Fragmentation, page 28

Example Configuring Class-Based Weighted Fair Queueing

The following example provides a sample configuration for Class-Based Weighted Fair Queueing (CBWFQ) with FRTS:

```
class-map voice
 match ip dscp ef
policy-map llq
 class voice
  priority 32
policy-map shape-policy-map
 class class-default
  shape average 64000
  shape adaptive 32000
  service-policy llq
map-class frame-relay shape-map-class

service-policy output shape-policy-map
interface serial 0/0
encapsulation frame-relay
interface serial 0/0.1 point-to-point
ip address 192.168.1.1 255.255.255.0
frame-relay interface-dlci 100
class shape-map-class
```

Example Configuring Class-Based Weighted Fair Queueing with Fragmentation

The following example provides a sample configuration for CBWFQ and fragmentation with FRTS. This configuration example is exactly the same as the example shown in the Example Configuring Class-Based Weighted Fair Queueing section, with the addition of the **frame-relay fragment** command to configure fragmentation.

```
class-map voice
 match ip dscp ef
policy-map llq
 class voice
  priority 32
policy-map shape-policy-map
 class class-default
  shape average 64000
  shape adaptive 32000
  service-policy llq
map-class frame-relay shape-map-class
 frame-relay fragment 80
 service-policy output shape-policy-map
interface serial 0/0
 encapsulation frame-relay
```

```
interface serial 0/0.1 point-to-point
 ip address 192.168.1.1 255.255.255.0
 frame-relay interface-dlci 100
  class shape-map-class
```

Example Backward Compatibility

The following configuration provides backward compatibility and interoperability with versions not compliant with RFC 1490. The **ietf** keyword is used to generate RFC 1490 traffic. 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 router to connect with a
! device running an older version of software
frame-relay map decnet 21.7 49 broadcast
```

Example Booting from a Network Server over Frame Relay

When booting from a TFTP server over Frame Relay, you cannot boot from a network server via a broadcast. You must boot from a specific TFTP host. Also, a **frame-relay map** command must exist for the host from which you will boot.

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. To boot over Frame Relay, you must explicitly give the address of the network server to boot from, and a **frame-relay map** entry must exist for that site. For example, if file "gs3-bfx.83-2.0" is to be booted from a host with IP address 131.108.126.111, the following commands must be in the configuration:

```
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 that can get to host 131.108.126.111.

The remote router must be configured with the following command:

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

This entry allows the remote router to return a boot image (from the network server) to the router booting over Frame Relay. Here, 101 is a DLCI of the router being booted.

Example Frame Relay Fragmentation Configuration

- Example FRF.12 Fragmentation, page 29

Example FRF.12 Fragmentation

The following example shows the configuration of pure end-to-end FRF.12 fragmentation and weighted fair queueing in the map class called "frag". The fragment payload size is set to 40 bytes. The "frag" map class is associated with DLCI 100 on serial interface 1.

```
router(config)#
interface serial 1
router(config-if)# frame-relay traffic-shaping
router(config-if)# frame-relay interface-dlci 100
router(config-fr-dlci)# class frag
router(config-fr-dlci)# exit
router(config)#
map-class frame-relay frag
router(config-map-class)# frame-relay cir 128000
router(config-map-class)# frame-relay bc 1280
router(config-map-class)# frame-relay fragment 40
router(config-map-class)# frame-relay fair-queue
```

Example TCP IP Header Compression

- Example IP Map with Inherited TCP IP Header Compression, page 30
- Example Using an IP Map to Override TCP IP Header Compression, page 30
- Example Disabling Inherited TCP IP Header Compression, page 31
- Example Disabling Explicit TCP IP Header Compression, page 31

Example IP Map with Inherited TCP IP Header Compression

Note Shut down the interface or subinterface prior to adding or changing compression techniques. Although shutdown is not required, shutting down the interface ensures that it is reset for the new data structures.

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)
```

This example also applies to dynamic mappings achieved with the use of Inverse ARP on point-to-point subinterfaces where no Frame Relay maps are configured.

Example Using an IP Map to Override TCP IP Header Compression

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:

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

Example Disabling Inherited TCP IP Header Compression

In this example, following is the initial configuration:

```
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
```

Enter the following commands to enable inherited TCP/IP header compression:

```
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. However, header compression is not disabled for the second map (DLCI 178), which is explicitly configured for header compression.

Example Disabling Explicit TCP IP Header Compression

In this example, the initial configuration is the same as in the preceding example, but you must enter the following set of commands to enable explicit TCP/IP header compression:

```
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 was explicitly configured for header compression.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS XE Wide-Area Networking configuration tasks	*Cisco IOS XE Wide-Area Networking Configuration Guide*, Release 2
Wide-Area networking commands	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Configuring Frame Relay

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1 Feature Information for Frame Relay

Feature Name	Releases	Feature Information
Frame Relay	Cisco IOS XE Release 2.1	Frame Relay is a high-performance WAN protocol that operates at the physical and data link layers.
Frame Relay Encapsulation	Cisco IOS XE Release 2.1	Frame Relay supports encapsulation of all supported protocols in conformance with RFC 1490, allowing interoperability between multiple vendors.

Feature Name	Releases	Feature Information
Frame Relay Fragmentation (FRF .12)	Cisco IOS XE Release 2.1	End-to-end FRF.12 fragmentation supports real-time and non-real-time data packets on lower-speed links without causing excessive delay to the real-time data. FRF. 12 fragmentation is defined by the FRF.12 Implementation Agreement.

Frame Relay Queueing and Fragmentation at the Interface

The Frame Relay Queueing and Fragmentation at the Interface feature introduces support for low-latency queueing (LLQ) and FRF.12 end-to-end fragmentation on a Frame Relay interface.

- Finding Feature Information, page 35
- Restrictions for Frame Relay Queueing and Fragmentation at the Interface, page 35
- Information About Frame Relay Queueing and Fragmentation at the Interface, page 36
- How to Configure Frame Relay Queueing and Fragmentation at the Interface, page 37
- Configuration Examples for Frame Relay Queueing and Fragmentation at the Interface, page 45
- Additional References, page 46
- Feature Information for Frame Relay Queueing and Fragmentation at the Interface, page 47

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Frame Relay Queueing and Fragmentation at the Interface

- Interface fragmentation and Frame Relay traffic shaping cannot be configured at the same time.
- Interface fragmentation and class-based fragmentation cannot be configured at the same time.
- Frame Relay switched virtual circuits (SVCs) are not supported.
- Hierarchical shaping and multiple shapers are not supported.

Information About Frame Relay Queueing and Fragmentation at the Interface

The Frame Relay Queueing and Fragmentation at the Interface feature simplifies the configuration of low-latency, low-jitter quality of service (QoS) by enabling the queueing policy and fragmentation configured on the main interface to apply to all permanent virtual circuits (PVCs) and subinterfaces under that interface. Before the introduction of this feature, queueing and fragmentation had to be configured on each individual PVC. Subrate shaping can also be configured on the interface.

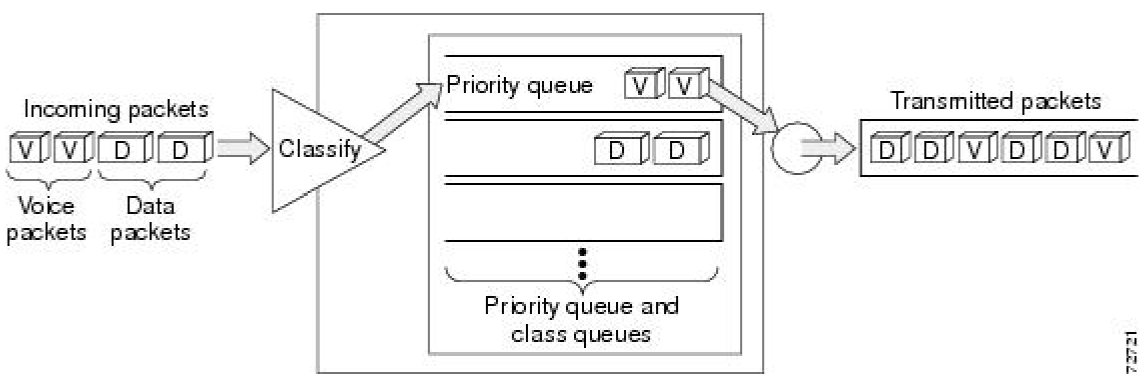
- How Frame Relay Queueing and Fragmentation at the Interface Works, page 36
- Benefits of Frame Relay Queueing and Fragmentation at the Interface, page 37

How Frame Relay Queueing and Fragmentation at the Interface Works

When FRF.12 end-to-end fragmentation is enabled on an interface, all PVCs on the main interface and its subinterfaces will have fragmentation enabled with the same configured fragment size. To maintain low latency and low jitter for high-priority traffic, the configured fragment size must be greater than the largest high-priority frames. This configuration will prevent high-priority traffic from being fragmented and queued behind lower-priority fragmented frames. If the size of a high-priority frame is larger than the configured fragment size, the high-priority frame will be fragmented. Local Management Interface (LMI) traffic will not be fragmented and is guaranteed its required bandwidth.

When a low-latency queueing policy map is applied to the interface, traffic through the interface is identified using class maps and is directed to the appropriate queue. Time-sensitive traffic such as voice should be classified as high priority and will be queued on the priority queue. Traffic that does not fall into one of the defined classes will be queued on the class-default queue. Frames from the priority queue and class queues are subject to fragmentation and interleaving. As long as the configured fragment size is larger than the high-priority frames, the priority queue traffic will not be fragmented and will be interleaved with fragmented frames from other class queues. This approach provides the highest QoS transmission for priority queue traffic. The figure below illustrates the interface queueing and fragmentation process.

Figure 3 *Frame Relay Queueing and Fragmentation at the Interface*



Subrate shaping can also be applied to the interface, but interleaving of high-priority frames will not work when shaping is configured. If shaping is not configured, each PVC will be allowed to send bursts of traffic up to the physical line rate.

When shaping is configured and traffic exceeds the rate at which the shaper can send frames, the traffic is queued at the shaping layer using fair queueing. After a frame passes through the shaper, the frame is queued at the interface using whatever queueing method is configured. If shaping is not configured, then queueing occurs only at the interface.

Note For interleaving to work, both fragmentation and the low-latency queueing policy must be configured with shaping disabled.

The Frame Relay Queueing and Fragmentation at the Interface feature supports the following functionality:

- Voice over Frame Relay
- Weighted Random Early Detection
- Frame Relay payload compression

Note When payload compression and Frame Relay fragmentation are used at the same time, payload compression is always performed before fragmentation.

- IP header compression

Benefits of Frame Relay Queueing and Fragmentation at the Interface

Simple Configuration

The Frame Relay Queueing and Fragmentation at the Interface feature allows fragmentation, low-latency queueing, and subrate shaping to be configured on a Frame Relay interface queue. The fragmentation and queueing and shaping policy will apply to all PVCs and subinterfaces under the main interface, eliminating the need to configure QoS on each PVC individually.

Flexible Bandwidth

This feature allows PVCs to preserve the logical separation of traffic from different services while reducing bandwidth partitioning between PVCs. Each PVC can send bursts of traffic up to the interface shaping rate or, if shaping is not configured, the physical interface line rate.

How to Configure Frame Relay Queueing and Fragmentation at the Interface

- Configuring Class Policy for the Priority Queue, page 38
- Configuring Class Policy for the Bandwidth Queues, page 39
- Configuring the Shaping Policy Using the Class-Default Class, page 40
- Configuring Queueing and Fragmentation on the Frame Relay Interface, page 41
- Verifying Frame Relay Queueing and Fragmentation at the Interface, page 43
- Monitoring and Maintaining Frame Relay Queueing and Fragmentation at the Interface, page 45

Configuring Class Policy for the Priority Queue

To configure a policy map for the priority class, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map*
4. **class** *class-name*
5. Router(config-pmap-c)# **priority** *bandwidth-kbps*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map* **Example:** `Router(config) policy-map policy1`	Specifies the name of the policy map to be created or modified. • Use this command to define the queueing policy for the priority queue.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class c1`	Specifies the name of a class to be created and included in the service policy. • The class name that you specify in the policy map defines the characteristics for that class and its match criteria as configured using the **class-map** command.
Step 5	Router(config-pmap-c)# **priority** *bandwidth-kbps* **Example:** `Router(config-pmap-c)# priority 30`	Creates a strict priority class and specifies the amount of bandwidth, in kbps, to be assigned to the class.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits the current configuration mode.

Configuring Class Policy for the Bandwidth Queues

To configure a policy map and create class policies that make up the service policy, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map*
4. **class** *class-name*
5. Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created or modified. • Use this command to define the queueing policy for the priority queue. • The bandwidth queues and the priority queue use the same policy map.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class c1`	Specifies the name of a class to be created and included in the service policy. • The class name that you specify in the policy map defines the characteristics for that class and its match criteria as configured using the **class-map**command.

	Command or Action	Purpose
Step 5	Router(config-pmap-c)# **bandwidth** *bandwidth-kbps* **Example:** `Router(config-pmap-c)# bandwidth 10`	Specifies the amount of bandwidth to be assigned to the class, in kbps, or as a percentage of the available bandwidth. Bandwidth must be specified in kbps or as a percentage consistently across classes. (Bandwidth of the priority queue must be specified in kbps.) • The sum of all bandwidth allocation on an interface cannot exceed 75 percent of the total available interface bandwidth. However, if you need to configure more than 75 percent of the interface bandwidth to classes, you can override the 75 percent maximum by using the **max-reserved-bandwidth** command.
Step 6	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits the current configuration mode.

Configuring the Shaping Policy Using the Class-Default Class

In general, the class-default class is used to classify traffic that does not fall into one of the defined classes. Even though the class-default class is predefined when you create the policy map, you still have to configure it. If a default class is not configured, traffic that does not match any of the configured classes is given best-effort treatment, which means that the network will deliver the traffic if it can, without any assurance of reliability, delay prevention, or throughput.

If you configure shaping in addition to queueing on the interface, use the class-default class to configure the shaping policy. The shaping policy will serve as the parent in a hierarchical traffic policy. The queueing policy will serve as the child policy. The class-default class is used for the shaping policy so that all traffic for the entire interface is shaped and a bandwidth-limited stream can be created.

To configure the shaping policy in the class-default class, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map*
4. **class class-default**
5. **shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]]
6. **service-policy** *policy-map-name*
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created or modified. • Use this command to define the shaping policy.
Step 4	**class class-default** **Example:** `Router(config-pmap)# class class-default`	Specifies the default class so that you can configure or modify its policy.
Step 5	**shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]] **Example:** `Router(config-pmap-c)# shape peak 10`	(Optional) Shapes traffic to the indicated bit rate according to the algorithm specified.
Step 6	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c)# service-policy policy1`	Specifies the name of a policy map to be used as a matching criterion (for nesting traffic policies [hierarchical traffic policies] within one another). • Use this command to attach the policy map for the priority queue (the child policy) to the shaping policy (the parent policy).
Step 7	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits the current configuration mode.

Configuring Queueing and Fragmentation on the Frame Relay Interface

To configure low-latency queueing and FRF.12 end-to-end fragmentation on a Frame Relay interface, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **interface** *type number*
2. Router(config-if)# **encapsulation frame-relay**
3. Router(config-if)# **frame-relay interface-dlci** *dlci*
4. Router(config-if-dlci)# **class** *name*
5. Router(config-if-dlci)# **exit**
6. Router(config)# **map-class frame-relay** *name*
7. Router(config-map-class)# **frame-relay fragment** *fragment-size* **end-to-end**
8. Router(config-map-class)# **no frame-relay adaptive-shaping**
9. Router(config-map-class)# **service-policy output** *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface** *type number*	Configures an interface type and enters interface configuration mode.
Step 2	Router(config-if)# **encapsulation frame-relay**	Enables Frame Relay encapsulation.
Step 3	Router(config-if)# **frame-relay interface-dlci** *dlci*	Assigns a DLCI to a specified Frame Relay subinterface on the router.
Step 4	Router(config-if-dlci)# **class** *name*	Associates a map class with a specified DLCI.
Step 5	Router(config-if-dlci)# **exit**	Exits configuration mode.
Step 6	Router(config)# **map-class frame-relay** *name*	Specifies a map class to define QoS values for a Frame Relay SVC or PVC.
Step 7	Router(config-map-class)# **frame-relay fragment** *fragment-size* **end-to-end**	Enables fragmentation of Frame Relay frames. • To maintain low latency and low jitter for priority queue traffic, configure the fragment size to be greater than the largest high-priority frame that would be expected.
Step 8	Router(config-map-class)# **no frame-relay adaptive-shaping**	Disables Frame Relay adaptive traffic shaping.
Step 9	Router(config-map-class)# **service-policy output** *policy-map-name*	Attaches a policy map to an output interface, to be used as the service policy for that interface. • If shaping is being used, use this command to attach the shaping policy (which includes the nested queueing policy) to the interface. • Interleaving of high-priority frames will not work if shaping is configured on the interface. • If shaping is not being used, use this command to attach the queueing policy to the interface.

Verifying Frame Relay Queueing and Fragmentation at the Interface

To verify the configuration and performance of Frame Relay queueing and fragmentation at the interface, perform the following steps:

SUMMARY STEPS

1. Enter the **show running-config** command to verify the configuration.
2. Enter the **show policy-map interface** command to display low-latency queueing information, packet counters, and statistics for the policy map applied to the interface. Compare the values in the "packets" and the "pkts matched" counters; under normal circumstances, the "packets" counter is much larger than the "pkts matched" counter. If the values of the two counters are nearly equal, then the interface is receiving a large number of process-switched packets or is heavily congested.
3. Enter the **show interfaces serial**command to display information about the queueing strategy, priority queue interleaving, and type of fragmentation configured on the interface. You can determine whether the interface has reached a congestion condition and packets have been queued by looking at the "Conversations" fields. A nonzero value for "max active" counter shows whether any queues have been active. If the "active" counter is a nonzero value, you can use the **show queue** command to view the contents of the queues.

DETAILED STEPS

Step 1 Enter the **show running-config** command to verify the configuration.

Example:

```
Router# show running-config
Building configuration...
.
.
.
class-map match-all voice
  match ip precedence 5
!
!policy-map llq
  class voice
    priority 64
policy-map shaper
  class class-default
   shape peak 96000
   service-policy llq
!
!interface Serial1/1
 ip address 16.0.0.1 255.255.255.0
 encapsulation frame-relay
 service-policy output shaper
 frame-relay fragment 80 end-to-end
!
```

Step 2 Enter the **show policy-map interface** command to display low-latency queueing information, packet counters, and statistics for the policy map applied to the interface. Compare the values in the "packets" and the "pkts matched" counters; under normal circumstances, the "packets" counter is much larger than the "pkts matched" counter. If the values of the two counters are nearly equal, then the interface is receiving a large number of process-switched packets or is heavily congested.
The following sample output for the **show policy-map interface command** is based on the configuration in Step 1:

Example:

```
Router# show policy-map interface serial 1/1
 Serial1/1
  Service-policy output:shaper
    Class-map:class-default (match-any)
      12617 packets, 1321846 bytes
      5 minute offered rate 33000 bps, drop rate 0 bps
      Match:any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           192000/96000     1992   7968      7968      83        1992
        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         12586     1321540   0         0         no
      Service-policy :llq
        Class-map:voice (match-all)
          3146 packets, 283140 bytes
          5 minute offered rate 7000 bps, drop rate 0 bps
          Match:ip precedence 1
          Weighted Fair Queueing
            Strict Priority
            Output Queue:Conversation 24
            Bandwidth 64 (kbps) Burst 1600 (Bytes)
            (pkts matched/bytes matched) 0/0
            (total drops/bytes drops) 0/0
        Class-map:class-default (match-any)
          9471 packets, 1038706 bytes
          5 minute offered rate 26000 bps
          Match:any
```

Step 3 Enter the **show interfaces serial**command to display information about the queueing strategy, priority queue interleaving, and type of fragmentation configured on the interface. You can determine whether the interface has reached a congestion condition and packets have been queued by looking at the "Conversations" fields. A nonzero value for "max active" counter shows whether any queues have been active. If the "active" counter is a nonzero value, you can use the **show queue** command to view the contents of the queues.

The following sample output for the **show interfaces serial**command is based on the configuration in Step 1:

Example:

```
Router# show interfaces serial 1/1
Serial1/1 is up, line protocol is up
  Hardware is M4T
  Internet address is 16.0.0.1/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 5/255, rxload 1/255
  Encapsulation FRAME-RELAY, crc 16, loopback not set
  Keepalive set (10 sec)
  Restart-Delay is 0 secs
  LMI enq sent  40, LMI stat recvd 40, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  Fragmentation type:end-to-end, size 80, PQ interleaves 0
  Broadcast queue 0/64, broadcasts sent/dropped 0/0, interface broadcasts 0
  Last input 00:00:03, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:06:34
  Input queue:0/75/0/0 (size/max/drops/flushes); Total output drops:0
  Queueing strategy:weighted fair
  Output queue:0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/1/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 33000 bits/sec, 40 packets/sec
     40 packets input, 576 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
```

```
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
15929 packets output, 1668870 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 output buffer failures, 0 output buffers swapped out
0 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Monitoring and Maintaining Frame Relay Queueing and Fragmentation at the Interface

To monitor and maintain Frame Relay queueing and fragmentation at the interface, use the following commands in privileged EXEC mode:

Command	Purpose
Router# **debug frame-relay fragment** [**event** | **interface** *type number dlci*]	Displays information related to Frame Relay fragmentation on a PVC.
Router# **show frame-relay fragment** [**interface** *type number* [*dlci*]]	Displays information about Frame Relay fragmentation.
Router# **show interfaces serial** *number*	Displays information about a serial interface.
Router# **show queue** *interface-type interface-number*	Displays the contents of packets inside a queue for a particular interface.
Router# **show policy-map interface** *number* [**input** | **output**]	Displays the packet statistics of all classes that are configured for all service policies on the specified interface.

Configuration Examples for Frame Relay Queueing and Fragmentation at the Interface

- Example Frame Relay Queueing Shaping and Fragmentation at the Interface, page 45
- Example Frame Relay Queueing and Fragmentation at the Interface, page 46

Example Frame Relay Queueing Shaping and Fragmentation at the Interface

The following example shows the configuration of a hierarchical policy for low-latency queueing, FRF.12 fragmentation, and shaping on serial interface 3/2. Note that traffic from the priority queue will not be interleaved with fragments from the class-default queue because shaping is configured.

```
class-map voice
 match access-group 101

policy-map llq
 class voice
```

```
  priority 64

policy-map shaper
 class class-default
  shape average 96000
  service-policy llq
interface serial 3/2
 ip address 10.0.0.1 255.0.0.0
 encapsulation frame-relay
 bandwidth 128
 clock rate 128000
 service-policy output shaper
 frame-relay fragment 80 end-to-end

 access-list 101 match ip any host 10.0.0.2
```

Example Frame Relay Queueing and Fragmentation at the Interface

The following example shows the configuration of low-latency queueing and FRF.12 fragmentation on serial interface 3/2. Because shaping is not being used, a hierarchical traffic policy is not needed and traffic from the priority queue will be interleaved with fragments from the other queues. Without shaping, the output rate of the interface is equal to the line rate or configured clock rate. In this example, the clock rate is 128,000 bps.

```
class-map voice
 match access-group 101

policy-map llq
 class voice
  priority 64
 class video
  bandwidth 32
interface serial 3/2
 ip address 10.0.0.1 255.0.0.0
 encapsulation frame-relay
 bandwidth 128
 clock rate 128000
 service-policy output llq
 frame-relay fragment 80 end-to-end
 access-list 101 match ip any host 10.0.0.2
```

Additional References

Related Documents

Related Topic	Document Title
WAN commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this functionality.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this functionality.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Frame Relay Queueing and Fragmentation at the Interface

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2 Feature Information for Frame Relay Queueing and Fragmentation at the Interface

Feature Name	Releases	Feature Information
Frame Relay Queueing and Fragmentation at the Interface	Cisco IOS XE Release 2.1	The Frame Relay Queueing and Fragmentation at the Interface feature introduces support for low-latency queueing (LLQ) and FRF.12 end-to-end fragmentation on a Frame Relay interface.

Frame Relay MIB Enhancements

The Cisco Frame Relay MIB describes managed objects that enable users to remotely monitor Frame Relay operations using Simple Network Management Protocol (SNMP). Frame Relay fragmentation is supported in the MIB.

- Finding Feature Information, page 49
- Prerequisites for Frame Relay MIB Enhancements, page 49
- Restrictions for Frame Relay MIB Enhancements, page 49
- Information About Frame Relay MIB Enhancements, page 50
- How to Configure Frame Relay MIB Enhancements, page 51
- Configuration Examples for Frame Relay MIB Enhancements, page 52
- Additional References, page 52
- Feature Information for Frame Relay MIB Enhancements, page 53

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Frame Relay MIB Enhancements

The tasks in this document assume that you have configured Frame Relay and SNMP on your devices.

To access the information introduced by the Frame Relay MIB enhancements, you must have the Cisco Frame Relay MIB in the MIB file called CISCO-FRAME-RELAY-MIB.my compiled in your network management system (NMS) application. You can find this MIB on the Web at Cisco's MIB website at

http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Restrictions for Frame Relay MIB Enhancements

- Frame Relay-ATM Network Interworking (FRF.5)
- Frame Relay-ATM Service Interworking (FRF.8)

- Frame Relay switching

Information About Frame Relay MIB Enhancements

- Feature Overview, page 50
- Benefits, page 51

Feature Overview

The Cisco Frame Relay MIB describes managed objects that enable users to remotely monitor Frame Relay operations using SNMP. The Frame Relay MIB Enhancements feature extends the Cisco Frame Relay MIB by adding MIB objects to monitor the following Frame Relay functionality:

- Frame Relay fragmentation
- Input and output rates of individual virtual circuits (VCs)

The table below describes the MIB tables and objects that are introduced by the Frame Relay MIB enhancements. For a complete description of the MIB, see the Cisco Frame Relay MIB file CISCO-FRAME-RELAY-MIB.my, available through Cisco.com at the following URL:

http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Table 3 MIB Tables and Objects Introduced by the Frame Relay MIB Enhancements

Table or Object	Description
cfrFragTable	Table of Frame Relay fragmentation information.
cfrFRF5ConnectionTable	Table of Frame Relay-ATM Network Interworking connection information.
cfrFRF8ConnectionTable	Table of Frame Relay-ATM Service Interworking connection information.
cfrSwitchingTable	Table of Frame Relay switching entries.
cfrExtCircuitTxDataRate	Average rate, in bytes per second, at which data is transmitted on a circuit.
cfrExtCircuitTxPktRate	Average number of packets sent per second on a circuit.
cfrExtCircuitRcvDataRate	Average rate, in bytes per second, at which data is received on a circuit.
cfrExtCircuitRcvPktRate	Average number of packets received per second on a circuit.

The Frame Relay MIB Enhancements feature also modifies the **load-interval** command to enable you to configure the load interval per permanent virtual circuit (PVC). The load interval is the length of time for which data is used to compute load statistics, including input rate in bits and packets per second, output rate in bits and packets per second, load, and reliability. Before the introduction of this feature, the load interval could be configured only for the interface.

Benefits

The Frame Relay MIB Enhancements enable you to use SNMP to monitor the following:

- Frame Relay fragmentation
- Frame Relay-ATM Network Interworking (FRF.5)
- Frame Relay-ATM Service Interworking (FRF.8)
- Frame Relay switching
- Input and output rates of individual virtual circuits (VCs)

How to Configure Frame Relay MIB Enhancements

- Setting the Load Interval for a PVC, page 51
- Verifying the Load Interval, page 51

Setting the Load Interval for a PVC

You can change the period of time over which a set of data is used for computing load statistics. Decisions, such as for dial backup, depend on these statistics. If you decrease the load interval, the average statistics are computed over a shorter period of time and are more responsive to bursts of traffic.

To change the length of time for which a set of data is used to compute load statistics for a PVC, use the following commands beginning in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **frame-relay interface-dlci** *dlci*
2. router(config-fr-dlci)# **load-interval** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay interface-dlci** *dlci*	Assigns a specific PVC to a DLCI[1], and enters Frame Relay DLCI configuration mode.
Step 2	router(config-fr-dlci)# **load-interval** *seconds*	Changes the length of time for which data is used to compute load statistics. The seconds argument must be a multiple of 30. The range is from 30 to 300 seconds. The default is 300 seconds.

Verifying the Load Interval

Use the **show running-config** command to verify that you have configured the load interval correctly.

1 DLCI = data-link connection identifier

Configuration Examples for Frame Relay MIB Enhancements

- Example Setting the Load Interval for a PVC, page 52

Example Setting the Load Interval for a PVC

In the following example, the load interval is set to 60 seconds for a Frame Relay PVC with the DLCI 100:

```
interface serial 1/1
 frame-relay interface-dlci 100
  load-interval 60
```

Additional References

Related Documents

Related Topic	Document Title
WAN commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this functionality.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this functionality.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Frame Relay MIB Enhancements

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4 Feature Information for Frame Relay MIB Enhancements

Feature Name	Releases	Feature Information
Frame Relay MIB Enhancements	Cisco IOS XE Release 2.1	The Cisco Frame Relay MIB describes managed objects that enable users to remotely monitor Frame Relay operations using SNMP.

Frame Relay PVC Interface Priority Queueing

The Frame Relay PVC Interface Priority Queueing feature provides an interface-level priority queueing scheme in which prioritization is based on destination permanent virtual circuit (PVC) rather than packet contents.

- Finding Feature Information, page 55
- Prerequisites for Frame Relay PVC Interface Priority Queueing, page 55
- Restrictions for Frame Relay PVC Interface Priority Queueing, page 55
- Information About Frame Relay PVC Interface Priority Queueing, page 56
- How to Configure Frame Relay PVC Interface Priority Queueing, page 57
- Configuration Examples for Frame Relay PVC Interface Priority Queuing, page 59
- Additional References, page 60
- Feature Information for Frame Relay PVC Interface Priority Queueing, page 61
- Glossary, page 62

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Frame Relay PVC Interface Priority Queueing

- PVCs should be configured to carry a single type of traffic.
- The network should be configured with adequate call admission control to prevent starvation of any of the priority queues.

Restrictions for Frame Relay PVC Interface Priority Queueing

- FR PIPQ is not supported on loopback or tunnel interfaces, or interfaces that explicitly disallow priority queueing.
- FR PIPQ is not supported with hardware compression.
- FR PIPQ cannot be enabled on an interface that is already configured with queueing other than FIFO queueing. FR PIPQ can be enabled if WFQ is configured, as long as WFQ is the default interface queueing method.

Information About Frame Relay PVC Interface Priority Queueing

- Feature Overview, page 56
- Benefits, page 57

Feature Overview

The FR PIPQ feature provides an interface-level priority queueing scheme in which prioritization is based on destination permanent virtual circuit (PVC) rather than packet contents. For example, FR PIPQ allows you to configure a PVC transporting voice traffic to have absolute priority over a PVC transporting signalling traffic, and a PVC transporting signalling traffic to have absolute priority over a PVC transporting data.

FR PIPQ provides four levels of priority: high, medium, normal, and low. The Frame Relay packet is examined at the interface for the data-link connection identifier (DLCI) value. The packet is then sent to the correct priority queue based on the priority level configured for that DLCI.

Note When using FR PIPQ, configure the network so that different types of traffic are transported on separate PVCs. FR PIPQ is not meant to be used when an individual PVC carries different traffic types that have different quality of service (QoS) requirements.

You assign priority to a PVC within a Frame Relay map class. All PVCs using or inheriting that map class will be classed according to the configured priority. If a PVC does not have a map class associated with it, or if the map class associated with it does not have priority explicitly configured, then the packets on that PVC will be queued on the default "normal" priority queue.

If you do not enable FR PIPQ on the interface using the **frame-relay interface-queue priority**command in interface configuration mode, configuring PVC priority within a map class will not be effective. At this time you have the option to also set the size (in maximum number of packets) of the four priority queues.

FR PIPQ works with or without Frame Relay traffic shaping (FRTS) and FRF.12. The interface-level priority queueing takes the place of the FIFO queueing or dual FIFO queueing normally used by FRTS and FRF.12. PVC priority assigned within FR PIPQ takes precedence over FRF.12 priority, which means that all packets destined for the same PVC will be queued on the same interface queue whether they were fragmented or not.

Note Although high priority PVCs most likely will transport only small packets of voice traffic, you may want to configure FRF.12 on these PVCs anyway to guard against any unexpectedly large packets.

Benefits

FR PIPQ provides four levels of PVC priority: high, medium, normal, and low. This method of queueing ensures that time/delay-sensitive traffic such as voice has absolute priority over signalling traffic, and that signalling traffic has absolute priority over data traffic, providing different PVCs are used for the different types of traffic.

How to Configure Frame Relay PVC Interface Priority Queueing

- Configuring PVC Priority in a Map Class, page 57
- Enabling FR PIPQ and Setting Queue Limits, page 57
- Assigning a Map Class to a PVC, page 58
- Verifying FR PIPQ, page 58
- Monitoring and Maintaining FR PIPQ, page 59

Configuring PVC Priority in a Map Class

To configure PVC priority within a map class, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **map-class frame-relay** *map-class-name*
2. Router(config-map-class)# **frame-relay interface-queue priority** {**high** | **medium**| **normal** | **low**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **map-class frame-relay** *map-class-name*	Specifies a Frame Relay map class.
Step 2	Router(config-map-class)# **frame-relay interface-queue priority** {**high** | **medium**| **normal** | **low**}	Assigns a PVC priority level to a Frame Relay map class.

Enabling FR PIPQ and Setting Queue Limits

To enable FR PIPQ and set the priority queue sizes, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Router(config)# **interface** *type number* [*name-tag*]
2. Router(config-if)# **encapsulation frame-relay**[**cisco** | **ietf**]
3. Router(config-if)# **frame-relay interface-queue priority** [*high-limit medium-limit normal-limit low-limit*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface** *type number* [*name-tag*]	Configures an interface type and enters interface configuration mode.
Step 2	Router(config-if)# **encapsulation frame-relay**[**cisco** | **ietf**]	Enables Frame Relay encapsulation.
Step 3	Router(config-if)# **frame-relay interface-queue priority** [*high-limit medium-limit normal-limit low-limit*]	Enables FR PIPQ and sets the priority queue limits.

Assigning a Map Class to a PVC

To assign a map class to a specific PVC, use the following commands beginning in interface configuration mode:

SUMMARY STEPS

1. Router(config-if)# **frame-relay interface-dlci** *dlci*
2. Router(config-fr-dlci)# **class** *map-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay interface-dlci** *dlci*	Specifies a single PVC on a Frame Relay interface.
Step 2	Router(config-fr-dlci)# **class** *map-class-name*	Associates a map class with a specified PVC.

Verifying FR PIPQ

To verify the configuration of FR PIPQ, use one or more of the following commands in privileged EXEC mode:

Command	Purpose
`Router#` **show frame-relay pvc** [**interface** *interface*][*dlci*]	Displays statistics about PVCs for Frame Relay interfaces.
`Router#` **show interfaces** [*type number*][*first*] [*last*]	Displays the statistical information specific to a serial interface.

Command	Purpose
Router# **show queueing** [**custom** | **fair** | **priority** | **random-detect** [**interface** *atm_subinterface* [**vc** [[*vpi*/] *vci*]]]]	Lists all or selected configured queueing strategies.

Monitoring and Maintaining FR PIPQ

To monitor and maintain FR PIPQ, use one or more of the following commands in privileged EXEC mode:

Command	Purpose
Router# **debug priority**	Debugs priority output queueing.
Router# **show frame-relay pvc** [**interface** *interface*][*dlci*]	Displays statistics about PVCs for Frame Relay interfaces.
Router# **show interfaces** [*type number*][*first*] [*last*]	Displays the statistical information specific to a serial interface.
Router# **show queue** *interface-name interface-number* [**vc** [*vpi*/] *vci*][*queue-number*]	Displays the contents of packets inside a queue for a particular interface or VC.
Router# **show queueing** [**custom** | **fair** | **priority** | **random-detect** [**interface** *atm_subinterface* [**vc** [[*vpi*/] vci]]]]	Lists all or selected configured queueing strategies.

Configuration Examples for Frame Relay PVC Interface Priority Queuing

- Monitoring and Maintaining FR PIPQ, page 59
- FR PIPQ Configuration Example, page 59

FR PIPQ Configuration Example

This example shows the configuration of four PVCs on serial interface 0. DLCI 100 is assigned high priority, DLCI 200 is assigned medium priority, DLCI 300 is assigned normal priority, and DLCI 400 is assigned low priority.

The following commands configure Frame Relay map classes with PVC priority levels:

```
Router(config)# map-class frame-relay HI
Router(config-map-class)# frame-relay interface-queue priority high
Router(config-map-class)# exit
Router(config)# map-class frame-relay MED
```

```
Router(config-map-class)# frame-relay interface-queue priority medium
Router(config-map-class)# exit
Router(config)# map-class frame-relay NORM
Router(config-map-class)# frame-relay interface-queue priority normal
Router(config-map-class)# exit
Router(config)# map-class frame-relay LOW
Router(config-map-class)# frame-relay interface-queue priority low
Router(config-map-class)# exit
```

The following commands enable Frame Relay encapsulation and FR PIPQ on serial interface 0. The sizes of the priority queues are set at a maximum of 20 packets for the high priority queue, 40 for the medium priority queue, 60 for the normal priority queue, and 80 for the low priority queue.

```
Router(config)# interface Serial0
Router(config-if)# encapsulation frame-relay
Router(config-if)# frame-relay interface-queue priority 20 40 60 80
```

The following commands assign priority to four PVCs by associating the DLCIs with the configured map classes:

```
Router(config-if)# frame-relay interface-dlci 100
Router(config-fr-dlci)# class HI
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 200
Router(config-fr-dlci)# class MED
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 300
Router(config-fr-dlci)# class NORM
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 400
Router(config-fr-dlci)# class LOW
Router(config-fr-dlci)# exit
```

Additional References

Related Documents

Related Topic	Document Title
WAN commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this functionality.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this functionality.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Frame Relay PVC Interface Priority Queueing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5 Feature Information for Frame Relay PVC Interface Priority Queueing

Feature Name	Releases	Feature Information
Frame Relay PVC Interface Priority Queueing	Cisco IOS XE Release 2.1	The FR PIPQ feature provides an interface-level priority queueing scheme in which prioritization is based on destination permanent virtual circuit (PVC) rather than packet contents. For example, FR PIPQ allows you to configure a PVC transporting voice traffic to have absolute priority over a PVC transporting signalling traffic, and a PVC transporting signalling traffic to have absolute priority over a PVC transporting data.

Glossary

DLCI --data-link connection identifier. Value that specifies a permanent virtual circuit (PVC) or switched virtual circuit (SVC) in a Frame Relay network.

FIFO queueing -- First-in, first-out queueing. FIFO involves buffering and forwarding of packets in the order of arrival. FIFO embodies no concept of priority or classes of traffic. There is only one queue, and all packets are treated equally. Packets are sent out an interface in the order in which they arrive.

Frame Relay traffic shaping --See FRTS.

FRF.12 --The FRF.12 Implementation Agreement was developed to allow long data frames to be fragmented into smaller pieces and interleaved with real-time frames. In this way, real-time voice and nonreal-time data frames can be carried together on lower-speed links without causing excessive delay to the real-time traffic.

FRTS --Frame Relay traffic shaping. FRTS uses queues on a Frame Relay network to limit surges that can cause congestion. Data is buffered and then sent into the network in regulated amounts to ensure that the traffic will fit within the promised traffic envelope for the particular connection.

PIPQ --Permanent virtual circuit (PVC) interface priority queueing. An interface-level priority queueing scheme in which prioritization is based on destination PVC rather than packet contents.

quality of service --Measure of performance for a transmission system that reflects its transmission quality and service availability.

WFQ --weighted fair queueing. Congestion management algorithm that identifies conversations (in the form of traffic streams), separates packets that belong to each conversation, and ensures that capacity is shared fairly among these individual conversations. WFQ is an automatic way of stabilizing network behavior during congestion and results in increased performance and reduced retransmission.

WRED --Weighted Random Early Detection. Combines IP Precedence and standard Random Early Detection (RED) to allow for preferential handling of voice traffic under congestion conditions without exacerbating the congestion. WRED uses and interprets IP Precedence to give priority to voice traffic over data traffic, dropping only data packets.

Frame Relay show Command and debug Command Enhancements

The Frame Relay show Command and debug Command Enhancements feature provides the ability to filter the output of certain Frame Relay **show** and **debug** commands on the basis of the interface and data-link connection identifier (DLCI). These enhancements facilitate network scalability and simplify network management and troubleshooting.

- Finding Feature Information, page 65
- Information About Frame Relay show Command and debug Command Enhancements, page 65
- Additional References, page 66
- Feature Information for Frame Relay show Command and debug Command Enhancements, page 67

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Frame Relay show Command and debug Command Enhancements

- Overview of the Frame Relay show Command and debug Command Enhancements, page 65
- Benefits of the Frame Relay Show Command and Debug Command Enhancements, page 66

Overview of the Frame Relay show Command and debug Command Enhancements

This feature introduces the following enhancments:

- The **show frame-relay map** command has been enhanced to allow map information to be displayed for specific interfaces and DLCIs.
- The **show frame-relay ip tcp header-compression** and **show frame-relay ip rtp header-compression** commands have been enhanced to allow header-compression information to be displayed for specific DLCIs.
- The **summary** keyword was added to the **show frame-relay pvc** command, allowing a summary of all PVCs on the system to be displayed.
- Conditional debugging support, which allows debug output to be filtered on the basis of interface and DLCI, was introduced for the following commands:
    - **debug frame-relay end-to-end**
    - **debug frame-relay events**
    - **debug frame-relay fragment**
    - **debug frame-relay fragment event**
    - **debug frame-relay ip**
    - **debug frame-relay ppp**
    - **debug frame-relay verbose**

Note Conditional debugging for Frame Relay **debug** commands is configured by using the **debug condition** command.

Benefits of the Frame Relay Show Command and Debug Command Enhancements

The Frame Relay show Command and debug Command Enhancements allow the output for some Frame Relay **show** commands and **debug** commands to be filtered on the basis of interface and DLCI. This enhancement saves network administrators time and frustration by eliminating the need to look through a large amount of output for information about a specific interface or DLCI. These enhancements can also reduce the amount of CPU processing time that is required to generate large amounts of **show** and **debug** output.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS XE Wide-Area Networking configuration tasks	*Cisco IOS XE Wide-Area Networking Configuration Guide*, Release 2
Wide-Area networking commands	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Frame Relay show Command and debug Command Enhancements

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 6 Feature Information for Frame Relay show Command and debug Command Enhancements

Feature Name	Releases	Feature Information
Frame Relay show Command and debug Command Enhancements	Cisco IOS XE Release 2.1	The Frame Relay show Command and debug Command Enhancements feature provides the ability to filter the output of certain Frame Relay **show** and **debug** commands on the basis of the interface and DLCI.