FRF .20 Support

First Published: June 19, 2006
Last Updated: November 17, 2006

The FRF .20 Support feature provides support for IP header compression (IPHC) over Frame Relay as described in the Frame Relay Forum Implementation Agreement FRF .20. Before the FRF .20 Support feature was introduced, Cisco IOS supported IPHC only on Cisco-encapsulated data-link connection identifiers (DLCIs) using a proprietary compression technique. This feature adds support for IPHC on IETF-encapsulated DLCIs.

This feature module describes how to configure FRF .20 IPHC over Frame Relay. This feature module also describes how to configure Enhanced Compressed Real-Time Transport Protocol (ECRTP).

Finding Feature Information in This Module

Your Cisco IOS software release may not support all of the features documented in this module. To reach links to specific feature documentation in this module and to see a list of the releases in which each feature is supported, use the "Feature Information for FRF .20 Support" section on page 36.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

CISCO SYSTEMS

Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Prerequisites for FRF .20 Support

You should understand the concepts and general configuration procedures for IP header compression. For information about IP header compression, see the Configuring Header Compression Using IPHC Profiles chapter of in Part 6 the *Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T.

Restrictions for FRF .20 Support

After a map class containing an IPHC profile is applied to an interface, subsequent attempts (using the **frame-relay ip rtp header-compression** command) to apply Real-time Transport Protocol (RTP) header compression for all Frame Relay maps on a physical interface are blocked.

Information About FRF .20 Support

To configure FRF .20 support, you should understand the following concept:

IP Header Compression

Header compression is a mechanism that compresses the IP header in a packet before the packet is transmitted. Header compression reduces network overhead and speeds up the transmission of either RTP or Transmission Control Protocol (TCP) packets. With IPHC over Frame Relay, compression parameters are negotiated across a DLCI.

One method of configuring header compression on your network is to use an IPHC profile. An IPHC profile is a kind of template within which you can configure the type of header compression that you want to use, set all of the optional features and parameters for header compression, and then apply the profile to an interface, subinterface, or Frame Relay permanent virtual circuit (PVC).

Enhanced Compressed Real-Time Transport Protocol

ECRTP is robust over links that are susceptible to frame loss. Header compression happens early in the Cisco Express Forwarding (CEF) switching path before queueing. If an interface or PVC is oversubscribed, all the dropped frames are compressed frames, which can impact CRTP compression performance. Using ECRTP can minimize the problem, because it provides better error recovery.

How to Configure FRF .20 Support

The following sections describe how to configure IPHC for FRF .20 over Frame Relay.

Configuring FRF .20 Support Using Profiles

To enable FRF .20 support on a DLCI, you attach an IPHC profile to a map class.

Restrictions

A map class containing a profile cannot overlap either an interface configuration or a PVC-level legacy configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **iphc-profile** *profile-name* **ietf**
4. **tcp**
5. **non-tcp contexts absolute** *number-of-contexts*
6. **rtp**
7. **class-map type traffic match-any** *class-map-name*
8. **match** [**ip**] **precedence** *precedence-value*
9. **map-class frame-relay** *map-class-name*
10. **iphc-profile** *profile-name*
11. **end**

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.

	Command or Action	Purpose
Step 3	**iphc-profile** *profile-name* **ietf** **Example:** `Router(config)# iphc-profile profile100 ietf`	Creates an IPHC profile and enter the IPHC-profile configuration mode. • *profile-name*—Name of the IPHC profile you are creating. The name can be a maximum of 40 alphanumeric characters. • **ietf**—IPHC profile type. Conforms with standards for FRF .20
Step 4	**tcp** **Example:** `Router(config-iphcp)# tcp`	(Optional) Enables TCP header compression.
Step 5	**non-tcp contexts absolute** *number-of-contexts* **Example:** `Router(config-iphcp)# non-tcp contexts absolute 60`	(Optional) Enables non-Transmission-Control-Protocol (TCP) header compression within an IPHC profile. This is required when RTP is configured.
Step 6	**rtp** **Example:** `Router(config-iphcp)# rtp`	(Optional) Enables RTP header compression within an IPHC profile.
Step 7	**class-map type traffic match-any** *class-map-name* **Example:** `Router(config-iphcp)# class-map type traffic match-any class100`	Creates a traffic class map, which is used for matching packets to a specified traffic class and enters class-map configuration mode. • **match-any**—Indicates that packets must meet one of the match criteria in order to be considered a member of the class. • *class-map-name*—Name of the class map.
Step 8	**match** [**ip**] **precedence** *precedence-value* **Example:** `Router(config-cmap)# match ip precedence 1`	Identifies the IP precedence values as match criteria. • **ip**—(Optional) Specifies that the match is for IPv4 packets only. If not used, the match is on both IP and IPv6 packets. • *precedence-value*—Specifies the exact value from 0 to 7 used to identify a precedence value.
Step 9	**map-class frame-relay** *map-class-name* **Example:** `Router(config-map-class)# compress header rtp`	Creates a Frame Relay map class and enters the static map class configuration mode.

	Command or Action	Purpose
Step 10	**iphc-profile** *profile-name* **Example:** `Router(config)# iphc-profile profile100 ietf`	Creates an IPHC profile. • *profile-name*—Name of the IPHC profile you are creating. The name can be a maximum of 40 alphanumeric characters.
Step 11	**end** **Example:** `Router(config-map-class)# end`	Ends the configuration session and returns to privileged EXEC mode.

Configuring an IPHC for Frame Relay Profile to a Map Class

You can use these procedures to configure an IPHC profile directly to a map class. You then apply the map class to the interface or DCLI directly. When the map class is applied, FRF .20 negotiation is started across the DCLI, and IPHC is started.

Prerequisite

Before FRF .20 negotiation can start, the PVC must be active.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **map-class frame-relay** *map-class-name*
4. **frame-relay iphc-profile** *profile-name*
5. **exit**
6. **frame-relay class** *class-name*
7. **frame-relay interface-dlci** *dlci*
8. **class** *class-name*
9. **end**

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.

	Command or Action	Purpose
Step 3	**map class frame-relay** *map-class-name* **Example:** `Router(config)# map class frame-relay mapclass100`	Creates a map class and enters static map class configuration mode. • *map-class-name*—Name of the map class. The name can be a maximum of 40 alphanumeric characters.
Step 4	**frame-relay iphc-profile** *profile-name* **Example:** `Router(config-map-class)# frame-relay iphc-profile profile100`	Attaches the IPHC profile to the Frame Relay map class. Enter the name of the IPHC profile that is to be attached to the Frame Relay map class created in Step 3.
Step 5	**exit** **Example:** `Router(config-map-class)# exit`	Exits static map class configuration mode.
Step 6	**frame-relay class** *class-name* **Example:** `Router(config)# frame-relay class frclass100`	Creates a Frame Relay class and enters Frame Relay class configuration mode.
Step 7	**frame-relay interface-dlci** *dlci* **Example:** `Router(config-fr-class)# frame-relay interface-dlci dlci 100`	Assigns a data-link connection identifier (DLCI) to a specified Frame Relay interface on the router or access server, and enters Frame Relay DLCI configuration mode. • *dlci*—DLCI number for the interface
Step 8	**class** *class-name* **Example:** `Router(config-fr-dlci)# class mapclass100`	Associates a map class with a specified DLCI. Enter the name of the map class to associate with the DLCI.
Step 9	**end** **Example:** `Router(config-pmap-c)# end`	Ends the configuration session and returns to privileged EXEC mode.

Configuring FRF .20 Support on VC Bundles

IPHC can be configured on virtual circuit (VC) bundles using the **map** command either on the bundle itself or on individual PVCs within the VC bundle using the **map-class** command.

You cannot enable IPHC in both places at the same time, because the map command is inherited by all the PVCs in the same bundle.

The following sections describe both methods of configuring FRF .20 support on VC bundles.

Configuring FRF .20 Support on VC Bundles Using the Map Command

To enable FRF .20 support on VC bundles using the **map** command, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **frame-relay map** *ip-address* **vc-bundle** *vc-bundle-name* [**rtp** | **tcp**] **header-compression**
4. **end**

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	**frame-relay map** *ip ip-address* **vc-bundle** *vc-bundle-name* [**rtp** | **tcp**] **header-compression** **Example:** `Router(config)# frame-relay map ip 209.165.200.230 vc-bundle test rtp header-compression`	Enables header compression on VC bundles.
Step 4	**end** **Example:** `Router(config)# end`	Ends the configuration session and returns to privileged EXEC mode.

Configuring FRF .20 Support on a PVC Using the Class Command

To enable FRF .20 support on a PVC using the **class** command, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **frame-relay vc-bundle** *vc-bundle-name*
4. **pvc** *name*
5. **class** *class-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**frame-relay vc-bundle** *vc-bundle-name* **Example:** `Router(config)#frame-relay vc-bundle vcbundle100`	Creates a Frame Relay permanent virtual circuit (PVC) bundle if it does not already exist, and enters Frame Relay VC-bundle configuration mode.
Step 3	**pvc** *name* **Example:** `Router(config fr-vcbundle)# frame-relay vc-bundle vcbundle100`	Creates or assigns a name to a permanent virtual circuit (PVC).
Step 4	**class** *class-name* **Example:** `Router(config-fr-vcbundle)# class mapclass100`	Associates a map class with a specified DLCI. Enter the name of the map class to associate with the DLCI.
Step 5	**end** **Example:** `Router(config-fr-vcbundle)# end`	Ends the configuration session and returns to privileged EXEC mode.

Enabling ECRTP

To enable ECRTP on an interface, use the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **iphc-profile** *profile-name* **ietf**
5. **non-tcp**
6. **rtp**
7. **recoverable-loss {dynamic |** *packet-drops***}**
8. **end**

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	**interface** *type number* Example: `Router(config)# interface serial 3/0`	Configures an interface type and enters interface configuration mode.
Step 4	**iphc-profile** *profile-name* **ietf** **Example:** `Router(config-if)# iphc-profile profile100 ietf`	Attaches an existing IPHC profile to an interface and enters the IPHC-profile configuration mode. • *profile-name*—Name of the IPHC profile you are attaching to the interface. **ietf**—IPHC profile type. Conforms with standards for FRF .20
Step 5	**non-tcp** **Example:** `Router(config-iphcp)# non-tcp`	Enables non-Transmission-Control-Protocol (TCP) header compression within an IPHC profile.
Step 6	**rtp** **Example:** `Router(config-iphcp)# rtp`	(Optional) Enables RTP header compression within an IPHC profile.
Step 7	**recoverable-loss** {**dynamic** | *packet-drops*} **Example:** `Router(config-iphcp)# recoverable-loss dynamic`	Enables ECRTP on an interface.
Step 8	**end** **Example:** `Router(config-iphcp)# end`	Ends the configuration session and returns to privileged EXEC mode.

Displaying Frame Relay Profile Status

You can display the current Frame Relay map entries and information about connections. To display active Frame Relay information, perform the following optional steps.

SUMMARY STEPS

1. **enable**
2. **show frame-relay map** [**interface** *type number*] [*dlci*]
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show frame-relay map** [**interface** *type number*] [*dlci*] **Example:** `Router# show frame-relay map 20`	Displays current Frame Relay map entries and information about connections. • **interface** *type number*—(Optional) Specifies an interface for which mapping information will be displayed. A space is optional between the interface type and number. • *dlci*—(Optional) Specifies a data-link connection identifier (DLCI) for which mapping information will be displayed. Range: 16 to 1022.
Step 3	**end** **Example:** `Router# end`	Ends the configuration session and returns to privileged EXEC mode.

Configuration Examples for FRF .20 Support

This section provides the following example:

- Adding an IPHC Profile to a Map Class, page 11

Adding an IPHC Profile to a Map Class

The following example shows how to add an IPHC profile to a Frame Relay map class. In this configuration, a profile is created using the **iphc-profile** command in global command mode. The profile is then attached to a Frame Relay map class.

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# iphc-profile iphc ietf
Router(config-iphcp)# tcp
Router(config-iphcp)# rtp
```

For non-TCP configuration, you must also configure the following:

```
Router(config-iphcp)# exit
Router(config)# map-class frame frf20
Router(config-map-class)# frame iphc
Router(config-map-class)# frame iphc-profile iphc
Router(config-map-class)# end
```

Additional References

The following sections provide references related to the FRF .20 Support feature.

Related Documents

Related Topic	Document Title
IPHC information and configuration tasks	*Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T
Additional QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*, Release 12.4T

Standards

Standard	Title
FRF .20	*Frame Relay IP Header Compression Implementation Agreement*

MIBs

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

RFCs

RFC	Title
RFC 3545	*Enhanced Compressed RPT for Links with High Delay, Packet Loss and Reordering*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Command Reference

This section contains only new and modified commands.

- **debug frame-relay ip tcp header-compression**
- **iphc-profile**
- **non-tcp**
- **recoverable-loss**
- **rtp**
- **show frame-relay ip rtp header-compression**
- **show frame-relay map**

debug frame-relay ip tcp header-compression

To display debugging information about TCP/IP header compression on Frame Relay interfaces, use the **debug frame-relay ip tcp header-compression** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay ip tcp header-compression

no debug frame-relay ip tcp header-compression

Syntax Description

This command has no arguments or keywords.

Defaults

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(9)T	This command was modified to display debugging output for control protocol frames for Frame Relay Forum Implementation Agreement (FRF) .20.
12.4(11)T	This command was modified to display debugging output for Enhanced Compressed Real-Time Transport Protocol (ECRTP).

Usage Guidelines

The **debug frame-relay ip tcp header-compression** command shows the control packets that are passed to initialize IP header compression (IPHC) on a permanent virtual circuit (PVC). For Cisco IPHC, typically two packets are passed: one sent and one received per PVC. (Inverse Address Resolution Protocol (InARP) packets are sent on PVCs that do not have a mapping defined between a destination protocol address and the data-link connection identifier (DLCI) or Frame Relay PVC bundle that connects to the destination address.) For FRF .20 IPHC, typically four packets are passed per PVC.

Debug messages are displayed only if the IPHC control protocol is renegotiated (for an interface or PVC state change or for a configuration change).

Examples

The following is sample output from the **debug frame-relay ip tcp header-compression** command when Cisco IPHC (not FRF .20 IPHC) is configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression

*Nov 14 09:22:07.991: InARP REQ: Tx compr_flags 43 *Nov 14 09:22:08.103: InARP RSP: Rx
compr_flags: 43
```

The following is sample output from the **debug frame-relay ip tcp header-compression** command when FRF .20 IPHC (without either Real-time Transport Protocol (RTP) or ECRTP) is configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression

FRF20(DLCI 16): Rxed Request, state 0
     : ident 0, tot len 19, conf_opts FE, len 15
       negotiation codes 1, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Txed Ack, state 0
     : ident 0, tot len 19, conf_opts FE, len 15
       negotiation codes 1, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Txed Request, state
0
     : ident 3, tot len 19, conf_opts FE, len 15
       negotiation codes 0, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Rxed Ack, state 2
     : ident 3, tot len 19, conf_opts FE, len 15
       negotiation codes 0, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 *Nov 14 09:18:37.019:
FRF20(DLCI 16): STARTING IPHC
```

The following is sample output from the **debug frame-relay ip tcp header-compression** command when FRF .20 IPHC and RTP are configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression

FRF20(DLCI 16): Txed Request, state 1
     : ident 0, tot len 21, conf_opts FE, len 17
       negotiation codes 1, version 1
  Par: IPV4, len 14, TCP_SPACE 16, NON_TCP_SPACE 16,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168
01:33:06: Subopt: rtp enabled
```

The following is sample output from the **debug frame-relay ip tcp header-compression** command when FRF .20 IPHC and ECRTP are configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression

FRF20(DLCI 16): Txed Request, state 1
     : ident 0, tot len 21, conf_opts FE, len 17
       negotiation codes 1, version 1
  Par: IPV4, len 14, TCP_SPACE 16, NON_TCP_SPACE 16,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168
01:33:06: Subopt: ecrtp enabled
```

Table 1 describes the significant fields shown in the displays.

Table 1 debug frame-relay ip tcp header-compression Field Descriptions

Field	Description
InARP REQ: Tx	Indicates that an InARP request was sent or received. Following are the possible values: • InARP REQ Tx—An InARP request was sent. • InARP REQ Rx—An InARP request was received.
InARP RSP: Rx	Indicates that an InARP response was sent or received. Following are the possible values: • InARP REQ Tx—An InARP response was sent. • InARP REQ Rx—An InARP response was received.
compr_flags: 43	Compression flags that Frame Relay peers use to negotiate Cisco IPHC options. It consists of a bit mask, and the number is displayed in hexadecimal format. Following are the bits: • 0x0001—TCP IPHC • 0x0002—RTP IPHC • 0x0004—Passive TCP compression • 0x0008—Passive RTP compression • 0x0040—Frame Relay IPHC options
FRF20(DLCI 16)	Indicates that the DLCI for this packet is configured with FRF .20 IPHC.
Txed Request	Direction of the IPHC control protocol message. Following are the possible values: • Txed Request • Txed Ack • Rxed Request • Rxed Ack Txed (transmitted) or Rxed (received) indicates the message direction, and Request or Ack (acknowledgement) indicates the message type. A peer sends a request indicating its configuration, and the other peer replies with an acknowledgement indicating its configuration. The lowest configuration value of this two-frame exchange sets the parameters in one direction. This means that typically four frames are exchanged in total: two Request/Ack pairs, with each pair negotiating the parameters in one direction.

Table 1 debug frame-relay ip tcp header-compression Field Descriptions (continued)

Field	Description
state 1	State of the FRF .20 IPHC protocol request. Following are the possible values: 0—FRF20_DISABLED. FRF .20 is disabled (because of an inactive PVC, an interface that is down, or a configuration mismatch). 1—FRF20_REQ_SENT. An FRF .20 control protocol request has been sent. 2—FRF20_REQ_RXED. An FRF .20 control protocol request has been received. 3—FRF20_WAIT_REQ. An FRF .20 control protocol request has been sent and acknowledged, and the local end is waiting for a request from the peer. 4—FRF20_OPERATIONAL. The FRF .20 control protocol is successfully negotiated, and frames can be compressed.
ident 0	Identifier. This is the transaction number used to correlate an FRF .20 control protocol request with an acknowledgement. This number is the same in messages that correspond to each other.
tot len 21	Sum (in bytes) of the lengths of the following: • All parameters • Negotiation codes • Identifier • Suboptions for each parameter set (IPV4 or IPV6)
conf_opts FE	Type of PPP parameter (expressed in hexadecimal). For FRF .20, the only possible value is FE (254 in decimal).
len 17	Total length of all parameters (in bytes).
negotiation codes 1	Negotiation state with the peer. Following are the possible values: • 0—Reply with response only. • 1—Reply with response and initiate request. With a response only, sending a response frame completes the negotiation. With a response and initiate request, the local peer also must send a request.
version 1	Version of the FRF .20 control protocol.
Par	List of parameters and values.
IPV4	Datagram type. The value is always IPV4, because Cisco IPHC does not support IPv6.
len 14	Total length (in bytes) of all parameters starting with IP type and ending with associated suboptions (if any). The value is greater than or equal to 12 depending on the suboptions.
TCP_SPACE 16	Maximum value of a TCP context identifier (CID) in the space of context identifiers allocated for TCP. Range: 3–255. Default value: 16. A value of zero means that TCP headers are not being compressed.

Table 1 debug frame-relay ip tcp header-compression Field Descriptions (continued)

Field	Description
NON_TCP_SPACE 16	Maximum value of a context identifier (CID) in the space of context identifiers allocated for non-TCP. Range: 3–1000. Cisco routers do not support the maximum value (65535) of the FRF .20 specification. Default value: 16. A value of zero means that non-TCP headers are not being compressed. These context identifiers are carried in COMPRESSED_NON_TCP, COMPRESSED_UDP and COMPRESSED_RTP packet headers.
F_MAX_PERIOD 256	Largest number of compressed non-TCP headers that can be sent without sending a full header. Range: 1–65535. Default value: 256. A value of zero indicates infinity, which means that the number of consecutive COMPRESSED_NON_TCP headers is unlimited.
F_MAX_TIME 5	Maximum time interval (in seconds) between full (uncompressed) headers. Range: 1–255. Default value: 5. A value of zero indicates infinity (meaning that no full headers will be transmitted).
MAX_HEADER 168	Largest header size (in bytes) that can be compressed. Range: 60–168. Cisco routers do not support the full range of values (60–65535) of the FRF .20 specification. Default value: 168.
01:33:06	Timestamp of the debug command output.
Subopt	Compression suboptions that are enabled. The value is either rtp or ecrtp.

iphc-profile

To create an IP Header Compression (IPHC) profile and to enter IPHC-profile configuration mode, use the **iphc-profile** command in global configuration mode. To attach an existing IPHC profile to an interface or subinterface, use the **iphc-profile** command in interface configuration mode. To delete the IPHC profile, use the **no** form of this command.

iphc-profile *profile-name* {**ietf** | **van-jacobson**}

no iphc-profile *profile-name*

Syntax Description

profile-name	Name of the IPHC profile to be created or attached. The IPHC profile name can be a maximum of 32 characters.
ietf	Specifies that the IPHC profile is for Internet Engineering Task Force (IETF) header compression.
van-jacobson	Specifies that the IPHC profile is for Van Jacobson header compression.

Command Default

No IPHC profile is created or attached.

Command Modes

Global configuration (to create an IPHC profile)
Interface configuration (to attach an existing IPHC profile to an interface or subinterface)

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

The **iphc-profile** command creates an IPHC profile used for enabling header compression and enters IPHC-profile configuration mode (config-iphcp). An IPHC profile is a template within which you can configure the type of header compression that you want to use, enable any optional features and settings for header compression, and then apply the profile to an interface, a subinterface, or a Frame Relay permanent virtual circuit (PVC).

Specifying the IPHC Profile Type

When you create an IPHC profile, you must specify the IPHC profile type by using either the **ietf** keyword or the **van-jacobson** keyword. The IETF profile type conforms to and supports the standards established with RFC 2507, RFC 2508, RFC 3544, and RFC 3545 and is typically associated with non-TCP header compression (for example, RTP header compression). The Van Jacobson profile type conforms to and supports the standards established with RFC 1144 and is typically associated with TCP header compression.

Note If you are using Frame Relay encapsulation, you must specify the **ietf** keyword (not the **van-jacobson** keyword).

Considerations When Specifying the IPHC Profile Type

When specifying the IPHC profile type, consider whether you are compressing TCP traffic or non-TCP traffic (that is, RTP traffic). Also consider the header compression format capabilities of the remote network link that will receive traffic. The IPHC profile type that you specify directly affects the header compression format used on the remote network links to which the IPHC profile is applied. *Only* TCP traffic is compressed on remote network links using a Van Jacobson IPHC profile, whereas TCP *and/or* non-TCP traffic (for example, RTP traffic) is compressed on remote network links using an IETF IPHC profile.

Note The header compression format in use on the router that you are configuring and the header compression format in use on the remote network link must match.

Configurable Header Compression Features and Settings

The specific set of header compression features and settings that you can configure (that is, enable or modify) is determined by the IPHC profile type that you specify (either IETF or Van Jacobson) when you create the IPHC profile. Both sets are listed below.

If you specify Van Jacobson as the IPHC profile type, you can enable TCP header compression and set the number of TCP contexts. Table 2 lists each available Van Jacobson IPHC profile type header compression feature and setting and the command used to enable it.

Table 2 Van Jacobson IPHC Profile Type Header Compression Features and Settings

Command	Feature or Setting
tcp	Enables TCP header compression.
tcp contexts	Sets the number of contexts available for TCP header compression.

If you specify IETF as the IPHC profile type, you can enable non-TCP header compression (that is, RTP header compression), along with a number of additional features and settings. Table 3 lists each available IETF IPHC profile type header compression feature and setting and the command or commands used to enable it.

Table 3 IETF IPHC Profile Type Header Compression Features and Settings

Command	Feature or Setting
feedback	Enables the context-status feedback messages from the interface or link.
maximum header	Sets the maximum size of the compressed IP header.
non-tcp	Enables non-TCP header compression.
non-tcp contexts	Sets the number of contexts available for non-TCP header compression.
rtp	Enables RTP header compression.
recoverable-loss	Enables Enhanced Compressed Real-Time Transport Protocol (ECRTP) on an interface.
refresh max-period **refresh max-time** **refresh rtp**	Sets the context refresh (full-header refresh) options, such as the amount of time to wait before a full header is refreshed.
tcp	Enables TCP header compression.
tcp contexts	Sets the number of contexts available for TCP header compression.

For More Information About IPHC Profiles

For more information about using IPHC profiles to configure header compression, see the "Header Compression" module and the "Configuring Header Compression Using IPHC Profiles" module of the *Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T.

Examples

In the following example, an IPHC profile called profile1 is created, and the Van Jacobson IPHC profile type is specified.

```
Router> enable
Router# configure terminal
Router(config)# iphc-profile profile1 van-jacobson
Router(config-iphcp)# end
```

In the following example, a second IPHC profile called profile2 is created. For this IPHC profile, the IETF IPHC profile type is specified.

```
Router> enable
Router# configure terminal
Router(config)# iphc-profile profile2 ietf
Router(config-iphcp)# end
```

In the following example, an existing IPHC profile called profile2 is attached to serial interface 3/0. For this IPHC profile, the IPHC profile type (in this case, IETF) of profile2 is specified.

```
Router> enable
Router# configure terminal
Router(config)# interface serial 3/0
Router(config-if)# iphc-profile profile2 ietf
Router(config-iphcp)# end
```

Related Commands

Command	Description
feedback	Enables the context-status feedback messages from the interface or link.
maximum header	Specifies the maximum size of the compressed IP header.
non-tcp	Enables non-TCP header compression within an IPHC profile.
non-tcp contexts	Sets the number of contexts available for non-TCP header compression.
recoverable-loss	Enables ECRTP on an interface.
refresh max-period	Sets the number of packets sent between full-header refresh occurrences.
refresh max-time	Sets the amount of time to wait before a full-header refresh occurrence.
refresh rtp	Enables a context refresh occurrence for RTP header compression.
rtp	Enables RTP header compression within an IPHC profile.
show iphc-profile	Displays configuration information for one or more IPHC profiles.
tcp	Enables TCP header compression within an IPHC profile.
tcp contexts	Set the number of contexts available for TCP header compression.

non-tcp

To enable non-Transmission-Control-Protocol (non-TCP) header compression within an IP Header Compression (IPHC) profile, use the **non-tcp** command in IPHC-profile configuration mode. To disable non-TCP header compression within an IPHC profile, use the **no** form of this command.

non-tcp

no non-tcp

Syntax Description

This command has no arguments or keywords.

Command Default

Non-TCP header compression is enabled.

Command Modes

IPHC-profile configuration

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

Intended for Use with IPHC Profiles

The **non-tcp** command is intended for use as part of an IPHC profile. An IPHC profile is used to enable and configure header compression on a network. For more information about using IPHC profiles to configure header compression, see the "Header Compression" module and the "Configuring Header Compression Using IPHC Profiles" module of the *Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T.

Examples

The following example shows how to configure an IPHC profile called profile2. In this example, non-TCP header compression is configured.

```
Router> enable
Router# configure terminal
Router(config)# iphc-profile profile2 ietf
Router(config-iphcp)# non-tcp
Router(config-iphcp)# end
```

Related Commands

Command	Description
iphc-profile	Creates an IPHC profile.

recoverable-loss

To enable Enhanced Compressed Real-Time Transport Protocol (ECRTP), use the **recoverable-loss** command in IPHC-profile configuration mode. To disable ECRTP, use the **no** form of this command.

recoverable-loss {**dynamic** | *packet-drops*}

no recoverable-loss {**dynamic** | *packet-drops*}

Syntax Description

dynamic	Indicates that the dynamic recoverable loss calculation is used.
packet-drops	Maximum number of consecutive packet drops. Range is from 1 to 8.

Command Default

ECRTP is disabled.

Command Modes

IPHC-profile configuration

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.4(11)T	Support was added for Frame Relay encapsulation.

Usage Guidelines

The **recoverable-loss** command is part of the ECRTP feature.

ECRPT Functionality

ECRTP reduces corruption by managing the way the compressor updates the context information at the decompressor. The compressor sends updated context information periodically to keep the compressor and decompressor synchronized. By repeating the updates, the probability of context corruption because of packet loss is minimized.

The synchronization of context information between the compressor and the decompressor can be performed dynamically (by specifying the **dynamic** keyword) or whenever a specific number of packets are dropped (by using the *packet-drops* argument).

The number of packet drops represents the quality of the link between the hosts. The lower the number of packet drops, the higher the quality of the link between the hosts.

The packet drops value is maintained independently for each context and does not have to be the same for all contexts.

Note If you specify the number of packet drops with the *packet-drops* argument, the **recoverable-loss** command automatically enables ECRTP.

Intended for Use with IPHC Profiles

The **recoverable-loss** command is intended for use as part of an IP Header Compression (IPHC) profile. An IPHC profile is used to enable and configure header compression on a network. For more information about using IPHC profiles to configure header compression, see the "Header Compression" module and the "Configuring Header Compression Using IPHC Profiles" module of the *Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T.

Examples

The following example shows how to configure an IPHC profile called profile2. In this example, ECRTP is enabled with a maximum number of five consecutive packet drops.

```
Router> enable
Router# configure terminal
Router(config)# iphc-profile profile2 ietf
Router(config-iphcp)# recoverable-loss 5
Router(config-iphcp)# end
```

Related Commands

Command	Description
iphc-profile	Creates an IPHC profile.

rtp

To enable Real-Time Transport Protocol (RTP) header compression within an IP Header Compression (IPHC) profile, use the **rtp** command in IPHC-profile configuration mode. To disable RTP header compression within an IPHC profile, use the **no** form of this command.

rtp

no rtp

Syntax Description

This command has no arguments or keywords.

Command Default

RTP header compression is enabled.

Command Modes

IPHC-profile configuration

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

The **rtp** command enables RTP header compression and automatically enables non-TCP header compression (the equivalent of using the **non-tcp** command).

Intended for Use with IPHC Profiles

The **rtp** command is intended for use as part of an IP Header Compression (IPHC) profile. An IPHC profile is used to enable and configure header compression on a network. For more information about using IPHC profiles to configure header compression, see the "Header Compression" module and the "Configuring Header Compression Using IPHC Profiles" module of the *Cisco IOS Quality of Service Solutions Configuration Guide*, Release 12.4T.

Examples

The following example shows how to configure an IPHC profile called profile2. In this example, RTP header compression is configured.

```
Router> enable
Router# configure terminal
Router(config)# iphc-profile profile2 ietf
Router(config-iphcp)# rtp
Router(config-iphcp)# end
```

Related Commands

Command	Description
iphc-profile	Creates an IPHC profile.
non-tcp	Enables non-TCP header compression within an IPHC profile.

show frame-relay ip rtp header-compression

To display Frame Relay Real-Time Transport Protocol (RTP) header compression statistics, use the **show frame-relay ip rtp header-compression** command in user EXEC or privileged EXEC mode.

show frame-relay ip rtp header-compression [**interface** *type number*] [*dlci*]

Syntax Description

interface *type number*	(Optional) Specifies an interface for which information will be displayed. A space between the interface type and number is optional.
dlci	(Optional) Specifies a data-link connection identifier (DLCI) for which information will be displayed. The range is from 16 to 1022.

Command Default

RTP header compression statistics are displayed for all DLCIs on interfaces that have RTP header compression configured.

Command Modes

User EXEC
Privileged EXEC

Command History

Release	Modification
11.3	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T. The output for this command was modified to display RTP header compression statistics for Frame Relay permanent virtual circuit (PVC) bundles.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC, and the *dlci* argument was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.4(9)T	The *dlci* argument was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	The output for this command was modified to display Enhanced Compressed Real-Time Transport Protocol (ECRTP) header compression statistics for Frame Relay permanent virtual circuit (PVC) bundles.

Examples

The following is sample output from the **show frame-relay ip rtp header-compression** command:

```
Router# show frame-relay ip rtp header-compression

 DLCI 21         Link/Destination info: ip 10.1.4.1
  Interface Serial3/0 DLCI 21 (compression on, Cisco)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

```
DLCI 20         Link/Destination info: ip 10.1.1.1
 Interface Serial3/1 DLCI 20 (compression on, Cisco)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 256 rx slots, 256 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

DLCI 21         Link/Destination info: ip 10.1.2.1
 Interface Serial3/1 DLCI 21 (compression on, Cisco)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 256 rx slots, 256 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

DLCI 22         Link/Destination info: ip 10.1.3.1
 Interface Serial3/1 DLCI 22 (compression on, Cisco)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 256 rx slots, 256 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

The following is sample output from the **show frame-relay ip rtp header-compression** command when ECRTP is enabled:

```
Router# show frame-relay ip rtp header-compression

DLCI 16         Link/Destination info: ip 10.0.0.1
 Interface Serial4/1 DLCI 16 (compression on, IETF, ECRTP)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 16 rx slots, 16 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 16 free contexts
```

In the following example, the **show frame-relay ip rtp header-compression** command displays information about DLCI 21:

```
Router# show frame-relay ip rtp header-compression 21

DLCI 21         Link/Destination info: ip 10.1.4.1
 Interface Serial3/0 DLCI 21 (compression on, Cisco)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 256 rx slots, 256 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

DLCI 21         Link/Destination info: ip 10.1.2.1
 Interface Serial3/1 DLCI 21 (compression on, Cisco)
   Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
            0 dropped, 0 buffer copies, 0 buffer failures
   Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
            0 bytes saved, 0 bytes sent
   Connect: 256 rx slots, 256 tx slots,
            0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

In the following example, the **show frame-relay ip rtp header-compression** command displays information for all DLCIs on serial interface 3/1:

```
Router# show frame-relay ip rtp header-compression interface serial3/1

 DLCI 20         Link/Destination info: ip 10.1.1.1
  Interface Serial3/1 DLCI 20 (compression on, Cisco)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

 DLCI 21         Link/Destination info: ip 10.1.2.1
  Interface Serial3/1 DLCI 21 (compression on, Cisco)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

 DLCI 22         Link/Destination info: ip 10.1.3.1
  Interface Serial3/1 DLCI 22 (compression on, Cisco)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

In the following example, the **show frame-relay ip rtp header-compression** command displays information only for DLCI 21 on serial interface 3/1:

```
Router# show frame-relay ip rtp header-compression interface serial3/1 21

 DLCI 21         Link/Destination info: ip 10.1.2.1
  Interface Serial3/1 DLCI 21 (compression on, Cisco)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

The following sample output from the **show frame-relay ip rtp header-compression** command shows statistics for a PVC bundle called MP-3-static:

```
Router# show frame-relay ip rtp header-compression interface Serial1/4

 vc-bundle MP-3-static      Link/Destination info:ip 10.1.1.1
  Interface Serial1/4:
    Rcvd:   14 total, 13 compressed, 0 errors
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:   15 total, 14 compressed,
             474 bytes saved, 119 bytes sent
             4.98 efficiency improvement factor
    Connect:256 rx slots, 256 tx slots,
             1 long searches, 1 misses 0 collisions, 0 negative cache hits
             93% hit ratio, five minute miss rate 0 misses/sec, 0 max
```

Table 4 describes the significant fields shown in the displays.

Table 4 show frame-relay ip rtp header-compression Field Descriptions

Field	Description
Interface	Type and number of the interface and type of header compression.
Rcvd:	Table of details concerning received packets.
total	Number of packets received on the interface.
compressed	Number of packets with compressed headers.
errors	Number of errors.
dropped	Number of dropped packets.
buffer copies	Number of buffers that were copied.
buffer failures	Number of failures in allocating buffers.
Sent:	Table of details concerning sent packets.
total	Total number of packets sent.
compressed	Number of packets sent with compressed headers.
bytes saved	Total savings in bytes because of compression.
bytes sent	Total bytes sent after compression.
efficiency improvement factor	Compression efficiency.
Connect:	Table of details about the connections.
rx slots	Total number of receive slots.
tx slots	Total number of transmit slots.
long searches	Searches that needed more than one lookup.
misses	Number of new states that were created.
hit ratio	Number of times that existing states were revised.
five minute miss rate	Average miss rate.
max	Maximum miss rate.

Related Commands

Command	Description
frame-relay ip rtp compression-connections	Specifies the maximum number of RTP header compression connections on a Frame Relay interface.
frame-relay ip rtp header-compression	Enables RTP header compression for all Frame Relay maps on a physical interface.
frame-relay map ip compress	Enables both RTP and TCP header compression on a link.
frame-relay map ip nocompress	Disables both RTP and TCP header compression on a link.
frame-relay map ip rtp header-compression	Enables RTP header compression per DLCI.
show ip rpf events	Displays RTP header compression statistics.

show frame-relay map

To display current Frame Relay map entries and information about connections, use the **show frame-relay map** command in privileged EXEC mode.

show frame-relay map [**interface** *type number*] [*dlci*]

Syntax Description

interface *type number*	(Optional) Specifies an interface for which mapping information will be displayed. A space is optional between the interface type and number.
dlci	(Optional) Specifies a data-link connection identifier (DLCI) for which mapping information will be displayed. Range: 16 to 1022.

Command Default

Static and dynamic Frame Relay map entries and information about connections for all DLCIs on all interfaces are displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.2(2)T	The display output for this command was modified to include the IPv6 address mappings of remote nodes to Frame Relay permanent virtual circuits (PVCs).
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(13)T	The display output for this command was modified to include information about Frame Relay PVC bundle maps.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB, the **interface** keyword was added, and the *dlci* argument was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(9)T	The **interface** keyword was added, and the *dlci* argument was added.

Examples

This section contains the following examples:

- Display All Maps or Maps for Specific DLCIs on Specific Interfaces or Subinterfaces: Example, page 31
- Display Maps for PVC Bundles: Example, page 32
- Display Maps for IPv6 Addresses: Example, page 33

Display All Maps or Maps for Specific DLCIs on Specific Interfaces or Subinterfaces: Example

The sample output in these examples uses the following configuration:

```
interface POS2/0
 no ip address
 encapsulation frame-relay
 frame-relay map ip 10.1.1.1 20 tcp header-compression
 frame-relay map ip 10.1.2.1 21 tcp header-compression
 frame-relay map ip 10.1.3.1 22 tcp header-compression
 frame-relay map bridge 23
 frame-relay interface-dlci 25
 frame-relay interface-dlci 26
 bridge-group 1
interface POS2/0.1 point-to-point
 frame-relay interface-dlci 24 protocol ip 10.1.4.1

interface Serial3/0
 no ip address
 encapsulation frame-relay
 serial restart-delay 0
 frame-relay map ip 172.16.3.1 20
 frame-relay map ip 172.16.4.1 21 tcp header-compression active
 frame-relay map ip 172.16.1.1 100
 frame-relay map ip 172.16.2.1 101
interface Serial3/0.1 multipoint
 frame-relay map ip 192.168.11.11 24
 frame-relay map ip 192.168.11.22 105
```

The following example shows how to display all maps:

```
Router# show frame-relay map

POS2/0 (up): ip 10.1.1.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): ip 10.1.2.1 dlci 21(0x15,0x450), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): ip 10.1.3.1 dlci 22(0x16,0x460), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): bridge dlci 23(0x17,0x470), static,
              CISCO, status deleted
POS2/0.1 (down): point-to-point dlci, dlci 24(0x18,0x480), broadcast
          status deleted
Serial3/0 (downup): ip 172.16.3.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
Serial3/0 (downup): ip 172.16.4.1 dlci 21(0x15,0x450), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
Serial3/0.1 (downup): ip 192.168.11.11 dlci 24(0x18,0x480), static,
              CISCO, status deleted
Serial3/0 (downup): ip 172.16.1.1 dlci 100(0x64,0x1840), static,
              CISCO, status deleted
Serial3/0 (downup): ip 172.16.2.1 dlci 101(0x65,0x1850), static,, CISCO,
              CISCO, status deleted
              ECRTP Header Compression (enabled, IETF), connections 16
              TCP/IP Header Compression (enabled, IETF), connections 16
Serial3/0.1 (downup): ip 192.168.11.22 dlci 105(0x69,0x1890), static,
              CISCO, status deleted
Serial4/0/1:0.1 (up): point-to-point dlci, dlci 102(0x66,0x1860), broadcast, CISCO
          status defined, active,
              RTP Header Compression (enabled), connections: 256
```

The following example shows how to display maps for a specific DLCI:

```
Router# show frame-relay map 20

POS2/0 (up): ip 10.1.1.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
Serial3/0 (down): ip 172.16.3.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
```

The following example shows how to display maps for a specific interface:

```
Router# show frame-relay map interface pos2/0

POS2/0 (up): ip 10.1.1.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): ip 10.1.2.1 dlci 21(0x15,0x450), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): ip 10.1.3.1 dlci 22(0x16,0x460), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
POS2/0 (up): bridge dlci 23(0x17,0x470), static,
              CISCO, status deleted
POS2/0.1 (down): point-to-point dlci, dlci 24(0x18,0x480), broadcast
          status deleted
```

The following example shows how to display maps for a specific DLCI on a specific interface:

```
Router# show frame-relay map interface pos2/0 20

POS2/0 (up): ip 10.1.1.1 dlci 20(0x14,0x440), static,
              CISCO, status deleted
              TCP/IP Header Compression (enabled), connections: 256
```

The following example shows how to display maps for a specific subinterface:

```
Router# show frame-relay map interface pos2/0.1

POS2/0.1 (down): point-to-point dlci, dlci 24(0x18,0x480), broadcast
          status deleted
```

The following example shows how to display maps for a specific DLCI on a specific subinterface:

```
Router# show frame-relay map interface pos2/0.1 24

POS2/0.1 (down): point-to-point dlci, dlci 24(0x18,0x480), broadcast
          status deleted
```

Display Maps for PVC Bundles: Example

The sample output in this example uses the following router configuration:

```
hostname router1
!
interface Serial2/0
 ip address 30.0.0.2 255.255.255.0
 encapsulation frame-relay
 frame-relay vc-bundle vcb1
  pvc 100 vcb1-classA
   precedence 1-7
   class vcb1-classA
  pvc 109 vcb1-others
   precedence other
```

```
   class others
 frame-relay intf-type dce
!
map-class frame-relay vcb1-classA
 frame-relay cir 128000
!
map-class frame-relay others
 frame-relay cir 64000

hostname router2
!
interface Serial3/3
 ip address 30.0.0.1 255.255.255.0
 encapsulation frame-relay
 frame-relay vc-bundle vcb1
  pvc 100 vcb1-classA
   precedence 1-7
   class vcb1-classA
  pvc 109 vcb1-others
   precedence other
   class others
!
map-class frame-relay vcb1-classA
 frame-relay cir 128000
!
map-class frame-relay others
 frame-relay cir 64000
```

The following sample output displays mapping information for two PVC bundles. The PVC bundle MAIN-1-static is configured with a static map. The map for PVC bundle MAIN-2-dynamic is created dynamically using Inverse Address Resolution Protocol (ARP).

```
Router# show frame-relay map

Serial1/4 (up): ip 10.1.1.1 vc-bundle MAIN-1-static, static,
          CISCO, status up
Serial1/4 (up): ip 10.1.1.2 vc-bundle MAIN-2-dynamic, dynamic,
          broadcast, status up
```

Display Maps for IPv6 Addresses: Example

The sample output in this example uses the following router configuration:

```
hostname router1
!
interface Serial2/0
 no ip address
 encapsulation frame-relay
!
interface Serial2/0.1 point-to-point
 ipv6 address 1::1/64
 frame-relay interface-dlci 101
!
interface Serial2/0.2 multipoint
 ipv6 address 2::1/64
 frame-relay map ipv6 2::2 201
 frame-relay interface-dlci 201
!

hostname router2
!
interface Serial3/3
 no ip address
 encapsulation frame-relay
```

```
 frame-relay intf-type dce
!
interface Serial3/3.1 point-to-point
 ipv6 address 1::2/64
 frame-relay interface-dlci 101
!
interface Serial3/3.2 multipoint
 ipv6 address 2::2/64
 frame-relay map ipv6 3::1 201
 frame-relay interface-dlci 201
!
```

The following sample output from the **show frame-relay map** command shows that the link-local and global IPv6 addresses (FE80::E0:F727:E400:A and 2001:0DB8:2222:1044::32; FE80::60:3E47:AC8:8 and 2001:0DB8:2222:1044::32) of two remote nodes are explicitly mapped to DLCI 17 and DLCI 19, respectively. Both DLCI 17 and DLCI 19 are terminated on interface serial 3 of this node; therefore, interface serial 3 of this node is a point-to-multipoint interface.

```
Router# show frame-relay map

Serial3 (up): ipv6 FE80::E0:F727:E400:A dlci 17(0x11,0x410), static,
              broadcast, CISCO, status defined, active
Serial3 (up): ipv6 2001:0DB8:2222:1044::32 dlci 19(0x13,0x430), static,
              CISCO, status defined, active

Serial3 (up): ipv6 2001:0DB8:2222:1044::32 dlci 17(0x11,0x410), static,
              CISCO, status defined, active
Serial3 (up): ipv6 FE80::60:3E47:AC8:8 dlci 19(0x13,0x430), static,
              broadcast, CISCO, status defined, active
```

Table 5 describes the significant fields shown in the displays.

Table 5 show frame-relay map Field Descriptions

Field	Description
POS2/0 (up)	Identifies a Frame Relay interface and its status (up or down).
ip 10.1.1.1	Destination IP address.
dlci 20(0x14,0x440)	DLCI that identifies the logical connection being used to reach this interface. This value is displayed in three ways: its decimal value (20), its hexadecimal value (0x14), and its value as it would appear on the wire (0x440).
vc-bundle	PVC bundle that serves as the logical connection being used to reach the interface.
static/dynamic	Indicates whether this is a static or dynamic entry.
broadcast	Indicates pseudobroadcasting.
CISCO	Indicates the encapsulation type for this map: either CISCO or IETF.

Table 5 show frame-relay map Field Descriptions (continued)

Field	Description
TCP/IP Header Compression (inherited), passive (inherited)	Indicates the header compression type (TCP/IP, Real-Time Transport Protocol (RTP), or Enhanced Compressed Real-Time Transport Protocol (ECRTP)) and whether the header compression characteristics were inherited from the interface or were explicitly configured for the IP map.
status defined, active	Indicates that the mapping between the destination address and the DLCI used to connect to the destination address is active.

Related Commands

Command	Description
show frame-relay pvc	Displays statistics about PVCs for Frame Relay interfaces.
show frame-relay vc-bundle	Displays attributes and other information about a Frame Relay PVC bundle.

Feature Information for FRF .20 Support

Table 6 lists the release history for this feature.

Not all commands may be available in your Cisco IOS software release. For release information about a specific command, see the command reference documentation.

Cisco IOS software images are specific to a Cisco IOS software release, a feature set, and a platform. Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.Table 6 lists only the Cisco IOS software release that introduced support for a given feature in a given Cisco IOS software release train. Unless noted otherwise, subsequent releases of that Cisco IOS software release train also support that feature.

Table 6 Feature Information for IP Header Compression Over Frame Relay

Feature Name	Releases	Feature Information
FRF .20 Support	12.4(9)T 12.4(11)T	This feature provides support for FRF .20. Support for IETF encapsulated DLCIs is also provided with this feature. In Cisco IOS Release 12.4(11)T, support for ECRTP was added, and the ability to enable IPHC on a specific PVC was added.

.