



# Distributed Multilink Frame Relay (FRF.16)

## Feature History

Release	Modification
12.0(24)S	The Distributed Multilink Frame Relay feature was introduced. This feature introduced Multilink Frame Relay (FRF.16) on VIP-enabled Cisco 7500 series routers.
12.3(4)T	This feature on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

This document describes the Distributed Multilink Frame Relay (dMFR) feature in Cisco IOS Release 12.0(24)S and Cisco IOS Release 12.3(4)T. The dMFR feature introduces MFR on VIP-enabled Cisco 7500 series routers. For information on MFR on other platforms, see the *Multilink Frame Relay (FRF.16)* document.

This document includes the following sections:

- [Feature Overview, page 1](#)
- [Supported Platforms, page 4](#)
- [Supported Standards, MIBs, and RFCs, page 5](#)
- [Prerequisites, page 5](#)
- [Configuration Tasks, page 5](#)
- [Monitoring and Maintaining Distributed Multilink Frame Relay, page 8](#)
- [Configuration Examples, page 9](#)
- [Command Reference, page 10](#)
- [Glossary, page 32](#)

## Feature Overview

The Distributed Multilink Frame Relay feature introduces functionality based on the Frame Relay Forum Multilink Frame Relay UNI/NNI Implementation Agreement (FRF.16) to VIP-enabled Cisco 7500 series routers. The Distributed Multilink Frame Relay feature provides a cost-effective way to increase bandwidth for particular applications by enabling multiple serial links to be aggregated into a single bundle of bandwidth. Multilink Frame Relay is supported on User-to-Network Interfaces (UNI) and Network-to-Network Interfaces (NNI) in Frame Relay networks.

## Multilink Frame Relay Bundles and Bundle Links

The Multilink Frame Relay feature enables you to create a virtual interface called a *bundle* or *bundle interface*. The bundle interface emulates a physical interface for the transport of frames. The Frame Relay data link runs on the bundle interface, and Frame Relay virtual circuits are built upon it.

The bundle is made up of multiple serial links, called *bundle links*. Each bundle link within a bundle corresponds to a physical interface. Bundle links are invisible to the Frame Relay data-link layer, so Frame Relay functionality cannot be configured on these interfaces. Regular Frame Relay functionality that you want to apply to these links must be configured on the bundle interface. Bundle links are visible to peer devices. The local router and peer devices exchange link integrity protocol control messages to determine which bundle links are operational and to synchronize which bundle links should be associated with which bundles.

## Link Integrity Protocol Control Messages

For link management, each end of a bundle link follows the MFR Link Integrity Protocol and exchanges link control messages with its peer (the other end of the bundle link). To bring up a bundle link, both ends of the link must complete an exchange of ADD\_LINK and ADD\_LINK\_ACK messages. To maintain the link, both ends periodically exchange HELLO and HELLO\_ACK messages. This exchange of hello messages and acknowledgments serve as a keepalive mechanism for the link. If a router is sending hello messages but not receiving acknowledgments, it will resend the hello message up to a configured maximum number of times. If the router exhausts the maximum number of retries, the bundle link line protocol is considered down (unoperational).

The bundle link interface's line protocol status is considered up (operational) when the peer device acknowledges that it will use the same link for the bundle. The line protocol remains up when the peer device acknowledges the hello messages from the local router.

The bundle interface's line status becomes up when at least one bundle link has its line protocol status up. The bundle interface's line status goes down when the last bundle link is no longer in the up state. This behavior complies with the class A bandwidth requirement defined in FRF.16.

The bundle interface's line protocol status is considered up when the Frame Relay data-link layer at the local router and peer device synchronize using the Local Management Interface (LMI), when LMI is enabled. The bundle line protocol remains up as long as the LMI keepalives are successful.

## Load Balancing

Distributed Multilink Frame Relay provides load balancing across the bundle links within a bundle. If a bundle link chosen for transmission happens to be busy transmitting a long packet, the load balancing mechanism can try another link, thus solving the problems seen when delay-sensitive packets have to wait.

## Benefits

### Flexible Pool of Bandwidth

By combining multiple physical interfaces into a bundle, you can design a Frame Relay interface with more bandwidth than is available from any single physical interface. For example, many new network applications require more bandwidth than is available on a T1 line. One option is to invest in a T3 line; however, T3 lines can be expensive and are not available in some locations. Distributed Multilink Frame Relay provides a cost-effective solution to this problem by allowing multiple T1 lines to be aggregated into a single bundle of bandwidth.

### Greater Service Resilience When Links Fail

Greater service resilience is provided when multiple physical interfaces are provisioned as a single bundle. When a link fails, the bundle continues to support the Frame Relay service by transmitting across the remaining bundle links.

## Restrictions

The Distributed Multilink Frame Relay feature has the following restrictions:

- ISDN interfaces and any type of virtual interfaces cannot be a bundle link.
- Distributed CEF is limited to IP traffic only; all other protocols are processed using the Route Switch Processor (RSP).
- Frame Relay fragmentation (FRF.12) is not supported.
- Multilink Frame Relay MIB (RFC 3020) is not supported.
- FRF.9 hardware compression over multilink Frame Relay is not supported.
- Each link in a bundle must reside on the same port adapter and all links in a bundle must have identical configurations. The same bandwidth for each link in the bundle is also recommended because bundles that contain individual links with different bandwidths process packets less efficiently.
- Fragmentation is not supported on the transmitting interface when used in conjunction with Distributed Multilink Frame Relay.
- The maximum differential delay is 50 ms.
- All T1 lines can be combined into one bundle. A VIP2-50 with 4 or 8 MB of SRAM supports up to 16 T1 bundles per VIP and a VIP2-50 with 2 MB of SRAM supports up to 8 T1 bundles per VIP. A maximum of 40 T1 bundles per VIP can be used on a VIP4-80.
- All E1 lines can be combined into one bundle. A VIP2-50 with 4 or 8 MB of SRAM supports up to 12 E1 bundles per VIP and a VIP2-50 with 2 MB of SRAM supports up to 8 E1 bundles per VIP. A maximum of 32 E1 bundles per VIP can be used on a VIP4-80.

## Related Documents

- [Cisco IOS Wide-Area Networking Configuration Guide](#), Release 12.3
- [Cisco IOS Wide-Area Networking Command Reference](#), Release 12.3
- [Multilink Frame Relay \(FRF.16\)](#) (provides information on nondistributed Multilink Frame Relay)
- [Cisco IOS Release 12.3 T command references](#) (information on 12.0 S and 12.3 T commands)

# Supported Platforms

- Cisco 7500 series router with a VIP2-50 or greater

This feature works on the following port adapters:

- PA-MC-T3
- PA-MC-2T3+
- PA-MC-E3
- PA-MC-2E1
- PA-MC-2T1
- PA-MC-4T1
- PA-MC-8T1
- PA-MC-8E1
- PA-MC-STM-1
- PA-MC-8TE1+
- PA-4T+
- PA-8T

## Determining Platform Support Through Cisco Feature Navigator

Cisco IOS software is packaged in feature sets that are supported on specific platforms. To get updated information regarding platform support for this feature, access Cisco Feature Navigator. Cisco Feature Navigator dynamically updates the list of supported platforms as new platform support is added for the feature.

Cisco Feature Navigator is a web-based tool that enables you to determine which Cisco IOS software images support a specific set of features and which features are supported in a specific Cisco IOS image. You can search by feature or release. Under the release section, you can compare releases side by side to display both the features unique to each software release and the features in common.

To access Cisco Feature Navigator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to [cco-locksmith@cisco.com](mailto:cco-locksmith@cisco.com). An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions found at this URL:

<http://www.cisco.com/register>

Cisco Feature Navigator is updated regularly when major Cisco IOS software releases and technology releases occur. For the most current information, go to the Cisco Feature Navigator home page at the following URL:

<http://www.cisco.com/go/fn>

## Availability of Cisco IOS Software Images

Platform support for particular Cisco IOS software releases is dependent on the availability of the software images for those platforms. Software images for some platforms may be deferred, delayed, or changed without prior notice. For updated information about platform support and availability of software images for each Cisco IOS software release, refer to the online release notes or, if supported, Cisco Feature Navigator.

# Supported Standards, MIBs, and RFCs

## Standards

*Multilink Frame Relay UNI/NNI Implementation Agreement* (FRF.16.1), July 2001

## MIBs

No new or modified MIBs are supported by this feature.

To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:

<http://tools.cisco.com/ITDIT/MIBS/servlet/index>

If Cisco MIB Locator does not support the MIB information that you need, you can also obtain a list of supported MIBs and download MIBs from the Cisco MIBs page at the following URL:

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

To access Cisco MIB Locator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to [cco-locksmith@cisco.com](mailto:cco-locksmith@cisco.com). An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions found at this URL:

<http://www.cisco.com/register>

## RFCs

No new or modified RFCs are supported by this feature.

# Prerequisites

- Distributed Cisco Express Forwarding (CEF) must be enabled globally.
- Multilink Frame Relay must be configured on the peer device.
- The multilink Frame Relay peer device must not send frames that require assembly.

# Configuration Tasks

See the following sections for configuration tasks for the Distributed Multilink Frame Relay feature. Each task in the list is identified as either optional or required.

- [Configuring a Multilink Frame Relay Bundle](#) (required)
- [Configuring a Multilink Frame Relay Bundle Link](#) (required)
- [Verifying Multilink Frame Relay](#) (optional)

## Configuring a Multilink Frame Relay Bundle

To configure the bundle interface for Distributed Multilink Frame Relay, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# <b>interface mfr</b> <i>number</i>	Configures a multilink Frame Relay bundle interface.
Step 2	Router(config-if)# <b>frame-relay multilink bid</b> <i>name</i>	(Optional) Assigns a bundle identification name to a multilink Frame Relay bundle.  <b>Note</b> The bundle identification (BID) will not go into effect until the interface has gone from the down state to the up state. One way to bring the interface down and back up again is by using the <b>shut</b> and <b>no shut</b> commands in interface configuration mode.

## Configuring a Multilink Frame Relay Bundle Link

To configure a bundle link interface for multilink Frame Relay, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# <b>interface serial</b> <i>number</i>	Selects a physical interface and enters interface configuration mode.
Step 2	Router(config-if)# <b>encapsulation frame-relay mfr</b> <i>number</i> [ <i>name</i> ]	Creates a multilink Frame Relay bundle link and associates the link with a bundle.  <b>Tips</b> To minimize latency that results from the arrival order of packets, we recommend bundling physical links of the same line speed in one bundle.
Step 3	Router(config-if)# <b>frame-relay multilink lid</b> <i>name</i>	(Optional) Assigns a bundle link identification name to a multilink Frame Relay bundle link.  <b>Note</b> The bundle link identification (LID) will not go into effect until the interface has gone from the down state to the up state. One way to bring the interface down and back up again is by using the <b>shut</b> and <b>no shut</b> commands in interface configuration mode.
Step 4	Router(config-if)# <b>frame-relay multilink hello</b> <i>seconds</i>	(Optional) Configures the interval at which a bundle link will send out hello messages. The default value is 10 seconds.

	Command	Purpose
Step 5	Router(config-if)# <b>frame-relay multilink ack</b> <i>seconds</i>	(Optional) Configures the number of seconds that a bundle link will wait for a hello message acknowledgment before resending the hello message. The default value is 4 seconds.
Step 6	Router(config-if)# <b>frame-relay multilink retry</b> <i>number</i>	(Optional) Configures the maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment. The default value is 2 tries.

## Verifying Multilink Frame Relay

To verify multilink Frame Relay configuration, use the **show frame-relay multilink** command.

The following example shows output for the **show frame-relay multilink** command. Because a particular bundle or bundle link is not specified, information for all bundles and bundle links is displayed.

```
Router# show frame-relay multilink

Bundle: MFR0, state up, class A, no fragmentation
  ID: Bundle-Dallas
  Serial5/1, state up/up, ID: BL-Dallas-1
  Serial5/3, state up/add-sent, ID: BL-Dallas-3

Bundle: MFR1, state down, class B, fragmentation
  ID: Bundle-NewYork#1
  Serial3/0, state up/up, ID: BL-NewYork-1
  Serial3/2, state admin-down/idle, ID: BL-NewYork-2
```

The following example shows output for the **show frame-relay multilink** command with the **serial number** option. It displays information about the specified bundle link.

```
Router# show frame-relay multilink serial3/2

Bundle links :
Serial3/2, HW state :Administratively down, Protocol state :Down_idle, LID :Serial3/2
Bundle interface = MFR0,  BID = MFR0
```

The following examples show output for the **show frame-relay multilink** command with the **serial number** and **detail** options. Detailed information about the specified bundle links is displayed. The first example shows a bundle link in the “idle” state. The second example shows a bundle link in the “up” state.

```
Router# show frame-relay multilink serial3 detail
Bundle links:

Serial3, HW state = up, link state = Idle, LID = Serial3
Bundle interface = MFR0,  BID = MFR0
  Cause code = none, Ack timer = 4, Hello timer = 10,
  Max retry count = 2, Current count = 0,
  Peer LID = Serial5/3, RTT = 0 ms
  Statistics:
  Add_link sent = 0, Add_link rcv'd = 10,
  Add_link ack sent = 0, Add_link ack rcv'd = 0,
  Add_link rej sent = 10, Add_link rej rcv'd = 0,
  Remove_link sent = 0, Remove_link rcv'd = 0,
  Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
```

```

Hello sent = 0, Hello rcv'd = 0,
Hello_ack sent = 0, Hello_ack rcv'd = 0,
outgoing pak dropped = 0, incoming pak dropped = 0

```

```

Router# show frame-relay multilink serial3 detail
Bundle links:

Serial3, HW state = up, link state = Up, LID = Serial3
Bundle interface = MFR0,  BID = MFR0
Cause code = none, Ack timer = 4, Hello timer = 10,
Max retry count = 2, Current count = 0,
Peer LID = Serial5/3, RTT = 4 ms
Statistics:
Add_link sent = 1, Add_link rcv'd = 20,
Add_link ack sent = 1, Add_link ack rcv'd = 1,
Add_link rej sent = 19, Add_link rej rcv'd = 0,
Remove_link sent = 0, Remove_link rcv'd = 0,
Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
Hello sent = 0, Hello rcv'd = 1,
Hello_ack sent = 1, Hello_ack rcv'd = 0,
outgoing pak dropped = 0, incoming pak dropped = 0

```

## Monitoring and Maintaining Distributed Multilink Frame Relay

To monitor and maintain Distributed Multilink Frame Relay, use one or more of the following commands in privileged EXEC mode:

Command	Purpose
Router# <b>debug frame-relay multilink</b> [ <b>control</b> [ <b>mfr number</b>   <b>serial number</b> ]]	Displays debug messages for multilink Frame Relay bundles and bundle links.
Router# <b>show frame-relay multilink</b> [ <b>mfr number</b>   <b>serial number</b> ] [ <b>detailed</b> ]	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.
Router# <b>show interfaces mfr number</b>	Displays information and packet statistics for the bundle interface.



# Configuration Examples

This section provides a Distributed Multilink Frame Relay configuration example.

## Distributed Multilink Frame Relay Configuration Example

The following example shows the configuration of bundle “MFR1”. Serial interfaces 5/0 and 6/0 are configured as bundle links.

```
interface MFR1
  frame-relay multilink bid first-bundle
  frame-relay traffic-shaping
  frame-relay class ocean

interface MFR1.1 point-to-point
  ip address 1.1.1.1 255.255.255.0
  frame-relay interface-dlci 100

interface Serial5/0
  encapsulation frame-relay MFR1
  frame-relay multilink lid first-link
  frame-relay multilink hello 9
  frame-relay multilink retry 3

interface Serial6/0
  encapsulation frame-relay MFR1
  frame-relay multilink ack 4
```

# Command Reference

This section documents new or modified commands. All other commands used with this feature are documented in the [Cisco IOS Release 12.3 T command reference](#) publications.

- [debug frame-relay multilink](#)
- [encapsulation frame-relay mfr](#)
- [frame-relay multilink ack](#)
- [frame-relay multilink bid](#)
- [frame-relay multilink hello](#)
- [frame-relay multilink lid](#)
- [frame-relay multilink retry](#)
- [interface mfr](#)
- [show frame-relay multilink](#)

# debug frame-relay multilink

To display debug messages for multilink Frame Relay bundles and bundle links, use the **debug frame-relay multilink** command in privileged EXEC mode. To disable the display of debug messages for multilink Frame Relay bundles and bundle links, use the **no** form of this command.

**debug frame-relay multilink** [**control** [*mfr number* | *serial number*]]


**no debug frame-relay multilink**

Syntax Description	<b>control</b>	(Optional) Displays incoming and outgoing bundle link control messages and bundle link status changes.
	<b>mfr number</b>	(Optional) Specific bundle interface for which information will be displayed.
	<b>serial number</b>	(Optional) Specific bundle link interface for which information will be displayed.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	<b>Release</b>	<b>Modification</b>
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

 **Caution** Using the **debug frame-relay multilink** command without the **control** keyword could severely impact router performance and is not recommended.

Using the **debug frame-relay multilink** without the **mfr** or **serial** keywords will display error conditions occurring at the bundle layer.

**Examples** The following example shows output from the **debug frame-relay multilink** command for bundle “MFR0”, which has 3 bundle links:

```
Router# debug frame-relay multilink control MFR0

00:42:54:Serial5/3(o):msg=Add_link, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3,
BL state=Idle
E1 00 01 01 07 4D 46 52 30 00
00:42:54:Serial5/2(o):msg=Add_link, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2,
BL state=Idle
E1 00 01 01 07 4D 46 52 30 00
```

```

00:42:54:Serial5/1(o):msg=Add_link, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1,
BL state=Idle
E1 00 01 01 07 4D 46 52 30 00
00:42:54:%LINK-3-UPDOWN:Interface MFR0, changed state to down
00:42:54:Serial5/3(i):msg=Add_link_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3,
BL state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:Serial5/2(i):msg=Add_link_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2,
BL state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:Serial5/1(i):msg=Add_link_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1,
BL state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:%SYS-5-CONFIG_I:Configured from console by console
00:43:00:Serial5/1(i):msg=Add_link, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1,
BL state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:00:Serial5/1(o):msg=Add_link_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1,
BL state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:00:%LINK-3-UPDOWN:Interface MFR0, changed state to up
00:43:00:Serial5/1(i):msg=Hello, Link=Serial5/1, Bundle=MFR0, Linkid=Serial5/1, BL
state=Up
E1 00 04 03 06 30 A7 E0 54 00
00:43:00:Serial5/1(o):msg=Hello_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:01:Serial5/2(i):msg=Add_link, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2,
BL state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:01:Serial5/2(o):msg=Add_link_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2,
BL state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:01:Serial5/2(i):msg=Hello, Link=Serial5/2, Bundle=MFR0, Linkid=Serial5/2, BL
state=Up
E1 00 04 03 06 30 A7 E0 54 00
00:43:01:Serial5/2(o):msg=Hello_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2,
BL state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:01:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/1, changed state to up
00:43:01:Serial5/3(i):msg=Add_link, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3,
BL state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:01:Serial5/3(o):msg=Add_link_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3,
BL state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:01:Serial5/3(i):msg=Hello, Link=Serial5/3, Bundle=MFR0, Linkid=Serial5/3, BL
state=Up
E1 00 04 03 06 30 A7 E0 54 00
00:43:01:Serial5/3(o):msg=Hello_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3,
BL state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:02:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/2 , changed state to up
00:43:02:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/3 , changed state to up

```

Table 1 describes the significant fields shown in the display.

**Table 1**     *debug frame-relay multilink Field Descriptions*

Field	Description
msg	Type of bundle link control message that was sent or received.
Link	Interface number of the bundle link.

**Table 1** *debug frame-relay multilink Field Descriptions (continued)*

Field	Description
Bundle	Bundle with which the link is associated.
Link id	Bundle link identification name.
BL state	Operational state of the bundle link.

**Related Commands**

Command	Description
<a href="#">show frame-relay multilink</a>	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

# encapsulation frame-relay mfr

To create a multilink Frame Relay bundle link and associate the link with a bundle, use the **encapsulation frame-relay mfr** command in interface configuration mode. To remove the bundle link from the bundle, use the **no** form of this command.

**encapsulation frame-relay mfr** *number* [*name*]

**no encapsulation frame-relay mfr**

## Syntax Description

<i>number</i>	The interface number of the multilink Frame Relay bundle with which this bundle link will be associated.
<i>name</i>	(Optional) Bundle link identification (LID) name. The name can be up to 49 characters long.

## Defaults

The default LID is the name of the physical interface.

## Command Modes

Interface configuration

## Command History

Release	Modification
12.0(17)S	This command was introduced on the Cisco 12000 series routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

## Usage Guidelines

Use the *name* argument to assign a LID name to the bundle link. This name will be used to identify the bundle link to peer devices and to enable the devices to determine which bundle links are associated with which bundles. The LID name can also be assigned or changed by using the **frame-relay multilink lid** command on the bundle link interface. If the LID name is not assigned, the default name is the name of the physical interface.



### Tips

To minimize latency that results from the arrival order of packets, we recommend bundling physical links of the same line speed in one bundle.

To remove a bundle link from a bundle, use the **no encapsulation frame-relay mfr** command or configure a new type of encapsulation on the interface by using the **encapsulation** command.

## Examples

The following example shows serial interface 0 being associated as a bundle link to bundle interface “mfr0”. The bundle link identification name is “BL1”.

```
interface mfr0
!
interface serial 0
 encapsulation frame-relay mfr0 BL1
```

## Related Commands

Command	Description
<a href="#">debug frame-relay multilink</a>	Displays debug messages for multilink Frame Relay bundles and bundle links.
<a href="#">frame-relay multilink lid</a>	Assigns a LID name to a multilink Frame Relay bundle link.
<a href="#">show frame-relay multilink</a>	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

# frame-relay multilink ack

To configure the number of seconds that a bundle link will wait for a hello message acknowledgment before resending the hello message, use the **frame-relay multilink ack** command in interface configuration mode. To reset this parameter to the default setting, use the **no** form of this command.

**frame-relay multilink ack** *seconds*

**no frame-relay multilink ack**

<b>Syntax Description</b>	<i>seconds</i>	Number of seconds that a bundle link will wait for a hello message acknowledgment before resending the hello message. The range is from 1 to 10 seconds.
---------------------------	----------------	--

<b>Defaults</b>	4 seconds
-----------------	-----------

<b>Command Modes</b>	Interface configuration
----------------------	-------------------------

<b>Command History</b>	Release	Modification
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

<b>Usage Guidelines</b>	The <b>frame-relay multilink ack</b> command can be configured only on bundle link interfaces that have been associated with a bundle using the <b>encapsulation frame-relay mfr</b> command.
	Both ends of a bundle link send out hello messages at regular intervals. When a peer device receives a hello message, it responds by sending an acknowledgment. This exchange of hello messages and acknowledgments serves as a keepalive mechanism for the link. If the bundle link sends a hello message but does not receive an acknowledgment, it will resend the hello message up to a configured maximum number of times. If the bundle link exhausts the maximum number of retries, the bundle link line protocol is considered down (unoperational).
	The <b>frame-relay multilink ack</b> command setting on the local router is independent of the setting on the peer device.

<b>Examples</b>	The following example shows how to configure the bundle link to wait 6 seconds before resending hello messages:
	<pre>interface serial0  encapsulation frame-relay mfr0  frame-relay multilink ack 6</pre>



Related Commands	Command	Description
	<a href="#">encapsulation frame-relay mfr</a>	Creates a multilink Frame Relay bundle link and associates the link with a bundle.
	<a href="#">frame-relay multilink hello</a>	Configures the interval at which a bundle link will send out hello messages.
	<a href="#">frame-relay multilink retry</a>	Configures the maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment.

# frame-relay multilink bid

To assign a bundle identification (BID) name to a multilink Frame Relay bundle, use the **frame-relay multilink bid** command in interface configuration mode. To reset the name to the default, use the **no** form of this command.

**frame-relay multilink bid** *name*

**no frame-relay multilink bid**

## Syntax Description

<i>name</i>	Bundle identification name (BID). The name can be up to 49 characters long.
-------------	---

## Defaults

The default BID is “mfr” plus the number assigned to the bundle using the **interface mfr** command; for example, “mfr0”.

## Command Modes

Interface configuration

## Command History

Release	Modification
12.0(17)S	This command was introduced.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

## Usage Guidelines

This command can be entered only at the multilink Frame Relay bundle interface.



### Note

You can enter the **frame-relay multilink bid** command at any time without affecting the current state of the interface; however, the BID will not go into effect until the interface has gone from the down state to the up state. One way to bring the interface down and back up again is by using the **shut** and **no shut** commands in interface configuration mode.

Only one BID is allowed per bundle. A later entry of the **frame-relay multilink bid** command supersedes prior entries.

The local and peer BIDs do not have to be unique.

## Examples

The following example shows how to assign a BID of “bundle1” to the multilink Frame Relay bundle. The previous BID for the bundle was “mfr0”.

```
interface mfr0
 frame-relay multilink bid bundle1
```

**Related Commands**

Command	Description
<a href="#">frame-relay multilink lid</a>	Assigns a LID name to a multilink Frame Relay bundle link.
<a href="#">interface mfr</a>	Configures a multilink Frame Relay bundle interface.
<a href="#">show frame-relay multilink</a>	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

# frame-relay multilink hello

To configure the interval at which a bundle link will send out hello messages, use the **frame-relay multilink hello** command in interface configuration mode. To reset this parameter to the default setting, use the **no** form of this command.

**frame-relay multilink hello** *seconds*

**no frame-relay multilink hello**

<b>Syntax Description</b>	<i>seconds</i>	Interval, in seconds, at which a bundle link will send out hello messages. The range is 1 through 180 seconds.
---------------------------	----------------	--

<b>Defaults</b>	10 seconds
-----------------	------------

<b>Command Modes</b>	Interface configuration
----------------------	-------------------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

**Usage Guidelines** The **frame-relay multilink hello** command can be configured only on bundle link interfaces that have been associated with a bundle using the **encapsulation frame-relay mfr** command.

Both ends of a bundle link send out hello messages at regular intervals. When a peer device receives a hello message, it responds by sending an acknowledgment. This exchange of hello messages and acknowledgments serves as a keepalive mechanism for the link. If the bundle link sends a hello message but does not receive an acknowledgment, it will resend the hello message up to a configured maximum number of times. If the bundle link exhausts the maximum number of retries, the bundle link line protocol is considered down (unoperational).

The setting of the hello message interval on the local router is independent of the setting on the peer device.

**Examples** The following example shows how to configure a bundle link to send hello messages every 15 seconds:

```
interface serial0
 encapsulation frame-relay mfr0
 frame-relay multilink hello 15
```

Related Commands	Command	Description
	<a href="#">encapsulation frame-relay mfr</a>	Creates a multilink Frame Relay bundle link and associates the link with a bundle.
	<a href="#">frame-relay multilink ack</a>	Configures the number of seconds that a bundle link will wait for a hello message acknowledgment before resending the hello message.
	<a href="#">frame-relay multilink retry</a>	Configures the maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment.

# frame-relay multilink lid

To assign a bundle link identification (LID) name to a multilink Frame Relay bundle link, use the **frame-relay multilink lid** command in interface configuration mode. To reset the name to the default, use the **no** form of this command.

**frame-relay multilink lid** *name*

**no frame-relay multilink lid**

<b>Syntax Description</b>	<i>name</i>	Bundle link identification name (LID). The name can be up to 49 characters long.
---------------------------	-------------	--

<b>Defaults</b>	The default LID is the name of the physical interface.
-----------------	--

<b>Command Modes</b>	Interface configuration
----------------------	-------------------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

<b>Usage Guidelines</b>	The <b>frame-relay multilink lid</b> command can be configured only on bundle link interfaces that have been associated with a bundle using the <b>encapsulation frame-relay</b> command with the <b>mfr</b> keyword.
-------------------------	---



<b>Note</b>	You can enter the <b>frame-relay multilink lid</b> command at any time without affecting the current state of the interface; however, the LID will not go into effect until the interface has gone from the down state to the up state. One way to bring the interface down and back up again is by using the <b>shut</b> and <b>no shut</b> commands in interface configuration mode.
-------------	--

The LID will be used to identify the bundle link to peer devices and to enable the devices to identify which bundle links are associated with which bundles. The LID can also be assigned when the bundle link is created by using the **encapsulation frame-relay** command with the *name* argument. If the LID is not assigned, the default LID is the name of the physical interface.

The local and peer LIDs do not need to be unique.

## Examples

The following example shows the LID “BL1” assigned to serial interface 0:

```
interface serial 0
 encapsulation frame-relay mfr0
 frame-relay multilink lid BL1
```

## Related Commands

Command	Description
<a href="#">encapsulation frame-relay mfr</a>	Creates a multilink Frame Relay bundle link and associates the link with a bundle.
<a href="#">frame-relay multilink bid</a>	Assigns a BID name to a multilink Frame Relay bundle.
<a href="#">show frame-relay multilink</a>	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

# frame-relay multilink retry

To configure the maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment, use the **frame-relay multilink retry** command in interface configuration mode. To reset this parameter to the default setting, use the **no** form of this command.

**frame-relay multilink retry** *number*

**no frame-relay multilink retry**

<b>Syntax Description</b>	<i>number</i>	Maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment. The range is 1 through 5 tries.
---------------------------	---------------	--

<b>Defaults</b>	2 tries
-----------------	---------

<b>Command Modes</b>	Interface configuration
----------------------	-------------------------

<b>Command History</b>	Release	Modification
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

<b>Usage Guidelines</b>	The <b>frame-relay multilink retry</b> command can be configured only on bundle link interfaces that have been associated with a bundle using the <b>encapsulation frame-relay</b> command with the <b>mfr</b> keyword.
	If the bundle link sends the maximum number of hello messages without receiving an acknowledgment, the bundle link line protocol is considered down (unoperational).
	The maximum number of retries configured on the local router is independent of the maximum number configured on the peer device.

<b>Examples</b>	The following example shows how to configure a bundle link to send a hello message a maximum of 3 times while waiting for an acknowledgment:
-----------------	--

```
interface serial0
 encapsulation frame-relay mfr0
 frame-relay multilink retry 3
```



Related Commands	Command	Description
	<a href="#">encapsulation frame-relay mfr</a>	Creates a multilink Frame Relay bundle link and associates the link with a bundle.
	<a href="#">frame-relay multilink ack</a>	Configures the number of seconds that a bundle link will wait for a hello message acknowledgment before resending the hello message.
	<a href="#">frame-relay multilink hello</a>	Configures the interval at which a bundle link will send out hello messages.

# interface mfr

To configure a multilink Frame Relay bundle interface, use the **interface mfr** command in global configuration mode. To remove the bundle interface, use the **no** form of this command.

**interface mfr** *number*

**no interface mfr** *number*

## Syntax Description

<i>number</i>	Number that will uniquely identify this bundle interface. The range is from 0 to 2,147,483,647.
---------------	---

## Defaults

A bundle interface is not configured.

Frame Relay encapsulation is the default encapsulation type for multilink Frame Relay bundle interfaces.

## Command Modes

Global configuration

## Command History

Release	Modification
12.0(17)S	This command was introduced.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

## Usage Guidelines

A bundle interface is a virtual interface that serves as the Frame Relay data link and performs the same functions as a physical interface. The bundle is made up of physical serial links, called bundle links. The bundle links within a bundle function as one physical link and one pool of bandwidth. Functionality that you want to apply to the bundle links must be configured on the bundle interface.

The **no interface mfr** command will work only if all bundle links have been removed from the bundle by using the **no encapsulation frame-relay mfr** command.

## Examples

The following example shows the configuration of a bundle interface called “mfr0”. The bundle identification (BID) name “BUNDLE-A” is assigned to the bundle. Serial interfaces 0 and 1 are assigned to the bundle as bundle links.

```
interface mfr0
  frame-relay multilink bid BUNDLE-A
!
interface serial0
  encapsulation frame-relay mfr0
!
interface serial1
  encapsulation frame-relay mfr0
```

## Related Commands

Command	Description
<a href="#">debug frame-relay multilink</a>	Displays debug messages for multilink Frame Relay bundles and bundle links.
<a href="#">encapsulation frame-relay mfr</a>	Creates a multilink Frame Relay bundle link and associates the link with a bundle.
<a href="#">frame-relay multilink bid</a>	Assigns a BID name to a multilink Frame Relay bundle.
<a href="#">show frame-relay multilink</a>	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

# show frame-relay multilink

To display configuration information and statistics about multilink Frame Relay bundles and bundle links, use the **show frame-relay multilink** command in privileged EXEC mode.

**show frame-relay multilink** [*mfr number* | *serial number*] [**detailed**]

Syntax Description	<b>mfr number</b>	(Optional) Specific bundle interface for which information will be displayed.
	<b>serial number</b>	(Optional) Specific bundle link interface for which information will be displayed.
	<b>detailed</b>	(Optional) More detailed information will be displayed, including counters for the control messages sent to and from the peer device and the status of the bundle links.

Command Modes	Privileged EXEC
---------------	-----------------

Command History	Release	Modification
	12.0(17)S	This command was introduced.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
	12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
	12.3(4)T	Support for this command on VIP-enabled Cisco 7500 series routers was integrated into Cisco IOS Release 12.3(4)T.

Usage Guidelines	If you do not specify a bundle or bundle link, information for all bundles and bundle links will be displayed.
------------------	--

## Examples

### All Bundles and Bundle Links Example

The following example shows output for the **show frame-relay multilink** command. Because a particular bundle or bundle link is not specified, information for all bundles and bundle links is displayed.

```
Router# show frame-relay multilink

Bundle:MFR0, State = Administratively down, class = A, fragmentation disabled
  BID = MFR0
  Bundle links :
    Serial2/1:3, HW state :up, Protocol state :Idle, LID :Serial2/1:3
    Serial2/1:2, HW state :up, Protocol state :Idle, LID :Serial2/1:2
    Serial2/1:1, HW state :up, Protocol state :Idle, LID :Serial2/1:1
```

### Bundle Link Example

The following example shows output for the **show frame-relay multilink** command with the **serial number** option. It displays information about the specified bundle link.

```
Router# show frame-relay multilink serial3/2
```

```
Bundle links :
Serial3/2, HW state :Administratively down, Protocol state :Down_idle, LID :Serial3/2
Bundle interface = MFR0,  BID = MFR0
```

### Detailed Bundle Link Examples

The following example show output for the **show frame-relay multilink** command with the **serial number** and **detailed** options. The example shows a bundle link in the “idle” state.

```
Router# show frame-relay multilink serial3 detail
Bundle links:
```

```
Serial3, HW state = up, link state = Idle, LID = Serial3
Bundle interface = MFR0,  BID = MFR0
Cause code = none, Ack timer = 4, Hello timer = 10,
Max retry count = 2, Current count = 0,
Peer LID = Serial5/3, RTT = 0 ms
Statistics:
Add_link sent = 0, Add_link rcv'd = 10,
Add_link ack sent = 0, Add_link ack rcv'd = 0,
Add_link rej sent = 10, Add_link rej rcv'd = 0,
Remove_link sent = 0, Remove_link rcv'd = 0,
Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
Hello sent = 0, Hello rcv'd = 0,
Hello_ack sent = 0, Hello_ack rcv'd = 0,
outgoing pak dropped = 0, incoming pak dropped = 0
```

The following example show output for the **show frame-relay multilink** command with the **serial number** and **detailed** options. The example shows a bundle link in the “up” state.

```
Router# show frame-relay multilink serial3 detail
Bundle links:
```

```
Serial3, HW state = up, link state = Up, LID = Serial3
Bundle interface = MFR0,  BID = MFR0
Cause code = none, Ack timer = 4, Hello timer = 10,
Max retry count = 2, Current count = 0,
Peer LID = Serial5/3, RTT = 4 ms
Statistics:
Add_link sent = 1, Add_link rcv'd = 20,
Add_link ack sent = 1, Add_link ack rcv'd = 1,
Add_link rej sent = 19, Add_link rej rcv'd = 0,
Remove_link sent = 0, Remove_link rcv'd = 0,
Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
Hello sent = 0, Hello rcv'd = 1,
Hello_ack sent = 1, Hello_ack rcv'd = 0,
outgoing pak dropped = 0, incoming pak dropped = 0
```

Table 2 describes the significant fields shown in the display.

**Table 2** *show frame-relay multilink Field Descriptions*

Field	Description
Bundle	Bundle interface.
State	Operational state of the bundle interface.
class	Class A indicates that if one bundle link is up, then the bundle is marked up, and all bundle links must be down before the bundle is marked down.
BID	Bundle identification.
Bundle links	Bundle links for which information will be displayed.
HW state	Operational state of the physical link.
Protocol state	Operational state of the bundle link line protocol.
Link state	Operational state of the bundle link.
LID	Bundle link identification.
Bundle interface	Bundle interface with which the bundle link is associated.
Cause code	May be one of the following values: <ul style="list-style-type: none"> <li>ack timer expiry—Add link synchronization process has been exhausted.</li> <li>bundle link idle—Peer's bundle link is idle. This usually occurs when the peer's bundle interface is shut.</li> <li>inconsistent bundle—Peer already has this bundle associated with a different bundle.</li> <li>loopback detected—Local bundle link's physical line is looped back.</li> <li>other—Indicates a LID mismatch, or that the ID length from the peer is too long, or that there has been a failure to allocate ID memory.</li> <li>unexpected Add_link—Add_link message is received when the bundle link is already in the up state. This code may appear when the line protocol is being set up, but will disappear once the connection has stabilized.</li> </ul>
Ack timer	Number of seconds the bundle link will wait for a hello acknowledgment before resending a hello message or resending an Add_link message used for initial synchronization.
Hello timer	Interval at which a bundle link sends out hello messages.
Max retry count	Maximum number of times a bundle link will resend a hello message before receiving an acknowledgment or resending an Add_link message.
Current count	Number of retries that have already been attempted.
Peer LID	Bundle link identification name of the peer end of the link.
RTT	Round trip time, as measured by utilizing the Timestamp Information Element in the HELLO and HELLO_ACK messages.

**Table 2**     *show frame-relay multilink Field Descriptions (continued)*

Field	Description
Statistics	Statistics for each bundle link will be displayed.
Add_link sent	Number of Add_link messages sent. Add_link messages notify the peer endpoint that the local endpoint is ready to process frames.
Add_link rcv'd	Number of Add_link messages received.
Add_link ack sent	Number of Add_link acknowledgments sent. Add_link acknowledgments notify the peer endpoint that an Add_link message was received.
Add_link ack rcv'd	Number of Add_link acknowledgments received.
Add_link rej sent	Number of Add_link_reject messages sent.
Add_link rej rcv'd	Number of Add_link_reject messages received.
Remove_link sent	Number of Remove_link messages sent. Remove_link messages notify the peer that on the local end a bundle link is being removed from the bundle.
Remove_link rcv'd	Number of Remove_link messages received.
Remove_link_ack sent	Number of Remove_link acknowledgments sent. Remove_link acknowledgments notify the peer that a Remove_link message has been received.
Remove_link_ack rcv'd	Number of Remove_link acknowledgments received.
Hello sent	Number of hello messages sent. Hello messages notify the peer endpoint that the local endpoint remains in the up state.
Hello rcv'd	Number of hello messages received.
Hello_ack sent	Number of hello acknowledgments sent. Hello acknowledgments notify the peer that hello messages have been received.
Hello_ack rcv'd	Number of hello acknowledgments received.
outgoing pak dropped	Number of outgoing packets dropped.
incoming pak dropped	Number of incoming packets dropped.

**Related Commands**

Command	Description
<a href="#">debug frame-relay multilink</a>	Displays debug messages for multilink Frame Relay bundles and bundle links.

# Glossary

**BID**—bundle identification. BID is the name used to identify the bundle. The BID can be assigned or the default can be used.

**bundle**—A logical grouping of one or more physical interfaces using the formats and procedures of multilink Frame Relay. A bundle emulates a physical interface to the Frame Relay data-link layer. The bundle is also referred to as the *mfr interface*.

**bundle link**—An individual physical interface that is a member of a bundle.

**DLCI**—data-link connection identifier. Value that identifies a permanent virtual circuit (PVC) in Frame Relay network.

**HELLO message**—Message that notifies a peer endpoint that the local endpoint is in the operational state (up).

**HELLO\_ACK**—Message that notifies a peer endpoint that a hello message has been received.

**LID**—link identification. LID is the name used to identify a bundle link. The LID can be assigned or the default can be used.

**LMI**—Local Management Interface. Set of enhancements to the basic Frame Relay specification. LMI includes support for a keepalive mechanism, which verifies that data is flowing; a multicast mechanism, which provides the network server with its local DLCI and the multicast DLCI; global addressing, which gives DLCIs global rather than local significance in Frame Relay networks; and a status mechanism, which provides an ongoing status report on the DLCIs known to the switch.

**NNI**—Network-to-Network Interface. The interface between two Frame Relay devices that are both located in a private network or both located in a public network.

**UNI**—User-to-Network Interface. The interface between a Frame Relay device in a public network and a Frame Relay device in a private network.