Configuring Virtual Profiles

First Published: December 15, 1997
Last Updated: November 20, 2014

A virtual profile is a unique application that can create and configure a virtual access interface dynamically when a dial-in call is received and that can tear down the interface dynamically when the call ends.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Configuring Virtual Profiles" section on page 936.

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

Contents

Prerequisites for Configuring Virtual Profiles

Cisco recommends that unnumbered addresses be used in virtual template interfaces to ensure that duplicate network addresses are not created on virtual access interfaces (VAIs).

Restrictions for Configuring Virtual Profiles

The **virtual-profile** command was removed from Cisco IOS Release 12.2(34)SB and 12.2(33)XNE, because Cisco 10000 series routers do not support the full VAIs these releases create and configuration errors could occur.

Information About Configuring Virtual Profiles

This section provides information about virtual profiles for use with virtual access interfaces and how virtual profiles work. Virtual profiles run on all Cisco IOS platforms that support Multilink PPP (MLP). Virtual profiles interoperate with Cisco dial-on-demand routing (DDR), MLP, and dialers such as ISDN. To configure virtual profiles, you should understand the following concepts:

- Virtual Profiles Overview, page 918
- How Virtual Profiles Work—Four Configuration Cases, page 920

Virtual Profiles Overview

Virtual profiles support these encapsulation methods:

- PPP
- MLP
- High-Level Data Link Control (HDLC)
- Link Access Procedure, Balanced (LAPB)
- X.25
- Frame Relay

Any commands for these encapsulations that can be configured under a serial interface can be configured under a virtual profile stored in a user file on an authentication, authorization, and accounting (AAA) server and a virtual profile virtual template configured locally. The AAA server daemon downloads them as text to the network access server and is able to handle multiple download attempts.

The configuration information for a virtual profiles virtual access interface can come from a virtual template interface or from user-specific configuration stored on a AAA server, or both.

If a B interface is bound by the calling line identification (CLID) to a created virtual access interface cloned from a virtual profile or a virtual template interface, only the configuration from the virtual profile or the virtual template takes effect. The configuration on the D interface is ignored unless successful binding occurs by PPP name. Both the link and network protocols run on the virtual access interface instead of the B channel, unless the encapsulation is PPP.

Moreover, in previous releases of Cisco IOS software, downloading a profile from an AAA server and creating and cloning a virtual access interface was always done after the PPP call answer and link control protocol (LCP) up processes. The AAA download is part of authorization. But in the current release, these operations must be performed before the call is answered and the link protocol goes up. This restriction is a new AAA nonauthenticated authorization step. The virtual profile code handles multiple download attempts and identifies whether a virtual access interface was cloned from a downloaded virtual profile.

When a successful download is done through nonauthenticated authorization and the configuration on the virtual profile has encapsulation PPP and PPP authentication, authentication is negotiated as a separate step after LCP comes up.

The per-user configuration feature also uses configuration information gained from a AAA server. However, per-user configuration uses *network* configurations (such as access lists and route filters) downloaded during Network Control Protocol (NCP) negotiations.

Two rules govern virtual access interface configuration by virtual profiles, virtual template interfaces, and AAA configurations:

- Each virtual access application can have at most one template to clone from but can have multiple AAA configurations to clone from (virtual profiles AAA information and AAA per-user configuration, which in turn might include configuration for multiple protocols).
- When virtual profiles are configured by virtual template, its template has higher priority than any other virtual template.

DDR Configuration of Physical Interfaces

Virtual profiles fully interoperate with physical interfaces in the following DDR configuration states when no other virtual access interface application is configured:

- Dialer profiles are configured for the interface—The dialer profile is used instead of the virtual profiles configuration.
- DDR is not configured on the interface—Virtual profiles overrides the current configuration.
- Legacy DDR is configured on the interface—Virtual profiles overrides the current configuration.

Note If a dialer interface is used (including any ISDN dialer), its configuration is used on the physical interface instead of the virtual profiles configuration.

Multilink PPP Effect on Virtual Access Interface Configuration

As shown in Table 1, exactly how a virtual access interface will be configured depends on the following three factors:

- Whether virtual profiles are configured by a virtual template, by AAA, by both, or by neither. In the table, these states are shown as “VP VT only,” “VP AAA only,” “VP VT and VP AAA,” and “No VP at all,” respectively.
- The presence or absence of a dialer interface.
- The presence or absence of MLP. The column label “MLP” is a stand-in for any virtual access feature that supports MLP and clones from a virtual template interface.

In Table 1, “(Multilink VT)” means that a virtual template interface is cloned *if* one is defined for MLP or a virtual access feature that uses MLP.

Table 1 Virtual Profiles Configuration Cloning Sequence

Virtual Profiles Configuration	MLP No Dialer	MLP Dialer	No MLP No Dialer	No MLP Dialer
VP VT only	VP VT	VP VT	VP VT	VP VT
VP AAA only	(Multilink VT) VP AAA	(Multilink VT) VP AAA	VP AAA	VP AAA
VP VT and VP AAA	VP VT VP AAA	VP VT VP AAA	VP VT VP AAA	VP VT VP AAA
No VP at all	(Multilink VT)[1]	Dialer[2]	No virtual access interface is created.	No virtual access interface is created.

1. The multilink bundle virtual access interface is created and uses the default settings for MLP or the relevant virtual access feature that uses MLP.
2. The multilink bundle virtual access interface is created and cloned from the dialer interface configuration.

The order of items in any cell of the table is important. Where VP VT is shown above VP AAA, it means that first the virtual profile virtual template is cloned on the interface, and then the AAA interface configuration for the user is applied to it. The user-specific AAA interface configuration adds to the configuration and overrides any conflicting physical interface or virtual template configuration commands.

Interoperability with Other Features That Use Virtual Templates

Virtual profiles also interoperate with virtual access applications that clone a virtual template interface. Each virtual access application can have at most one template to clone from but can clone from multiple AAA configurations.

The interaction between virtual profiles and other virtual template applications is as follows:

- If virtual profiles are enabled and a virtual template is defined for it, the virtual profile virtual template is used.
- If virtual profiles are configured by AAA alone (no virtual template is defined for virtual profiles), the virtual template for another virtual access application (virtual private dialup networks or VPDNs, for example) can be cloned onto the virtual access interface.
- A virtual template, if any, is cloned to a virtual access interface before the virtual profiles AAA configuration or AAA per-user configuration. AAA per-user configuration, if used, is applied last.

How Virtual Profiles Work—Four Configuration Cases

This section describes virtual profiles and the various ways that they can work with virtual template interfaces, user-specific AAA interface configuration, and MLP or another feature that requires MLP.

Virtual profiles separate configuration information into two logical parts:

- Generic—Common configuration for dial-in users plus other router-dependent configuration. This common and router-dependent information can define a virtual template interface stored locally on the router. The generic virtual template interface is independent of and can override the configuration of the physical interface on which a user dialed in.
- User-specific interface information—Interface configuration stored in a user file on an AAA server; for example, the authentication requirements and specific interface settings for a specific user. The settings are sent to the router in the response to the request from the router to authenticate the user, and the settings can override the generic configuration. This process is explained more in the section "Virtual Profiles Configured by AAA" later in this chapter.

These logical parts can be used separately or together. Four separate cases are possible:

- Case 1: Virtual Profiles Configured by Virtual Template, page 921—Applies the virtual template.
- Case 2: Virtual Profiles Configured by AAA, page 922—Applies the user-specific interface configuration received from the AAA server.
- Case 3: Virtual Profiles Configured by Virtual Template and AAA Configuration, page 922—Applies the virtual template and the user-specific interface configuration received from the AAA server.
- Case 4: Virtual Profiles Configured by AAA, and a Virtual Template Defined by Another Application, page 923—Applies the other application's virtual template interface and then applies the user-specific interface configuration received from the AAA server.

Note All cases assume that AAA is configured globally on the router, that the user has configuration information in the user file on the AAA server, that PPP authentication and authorization proceed as usual, and that the AAA server sends user-specific configuration information in the authorization approval response packet to the router.

The cases also assume that AAA works as designed and that the AAA server sends configuration information for the dial-in user to the router, even when virtual profiles by virtual template are configured.

Case 1: Virtual Profiles Configured by Virtual Template

In the case of virtual profiles configured by virtual template, the software functions as follows:

- If the physical interface is configured for dialer profiles (a DDR feature), the router looks for a dialer profile for the specific user.
- If a dialer profile is found, it is used instead of virtual profiles.
- If a dialer profile is not found for the user, or legacy DDR is configured, or DDR is not configured at all, virtual profiles create a virtual access interface for the user.

The router applies the configuration commands that are in the virtual template interface to create and configure the virtual profile. The template includes generic interface information and router-specific information, but no user-specific information. No matter whether a user dialed in on a synchronous serial, an asynchronous serial, or an ISDN interface, the dynamically created virtual profile for the user is configured as specified in the virtual template.

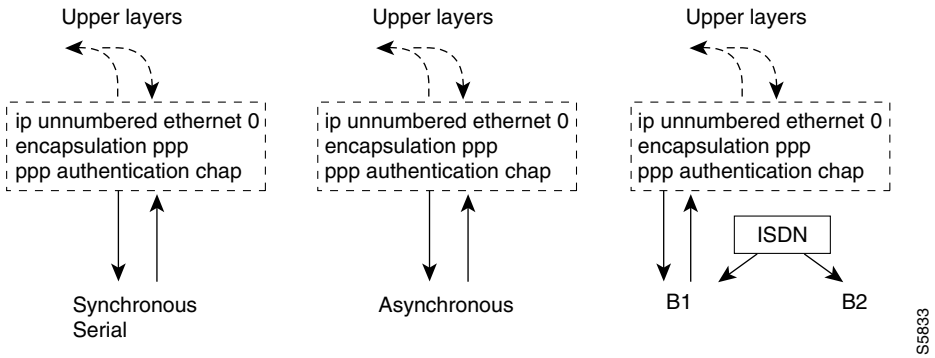
Then the router interprets the lines in the AAA authorization approval response from the server as Cisco IOS commands to apply to the virtual profile for the user.

Data flows through the virtual profile, and the higher layers treat it as the interface for the user.

For example, if a virtual template included only the three commands **ip unnumbered ethernet 0**, **encapsulation ppp,** and **ppp authentication chap**, the virtual profile for any dial-in user would include those three commands.

In Figure 1, the dotted box represents the virtual profile configured with the commands that are in the virtual template, no matter which interface the call arrives on.

Figure 1 Virtual Profiles by Virtual Template



See the "Configuring Virtual Profiles by Virtual Template" section on page 924 for configuration tasks for this case.

Case 2: Virtual Profiles Configured by AAA

In this case, no dialer profile (a DDR feature) is defined for the specific user and no virtual template for virtual profiles is defined, but virtual profiles by AAA are enabled on the router.

During the PPP authorization phase for the user, the AAA server responds as usual to the router. The authorization approval contains configuration information for the user. The router interprets each of the lines in the AAA response from the server as Cisco IOS commands to apply to the virtual profile for the user.

Note If MLP is negotiated, the MLP virtual template is cloned first (this is the second row), and then interface-specific commands included in the AAA response from the server for the user are applied. The MLP virtual template overrides any conflicting interface configuration, and the AAA interface configuration overrides any conflicting configuration from both the physical interface and the MLP virtual template.

The router applies all the user-specific interface commands received from the AAA server.

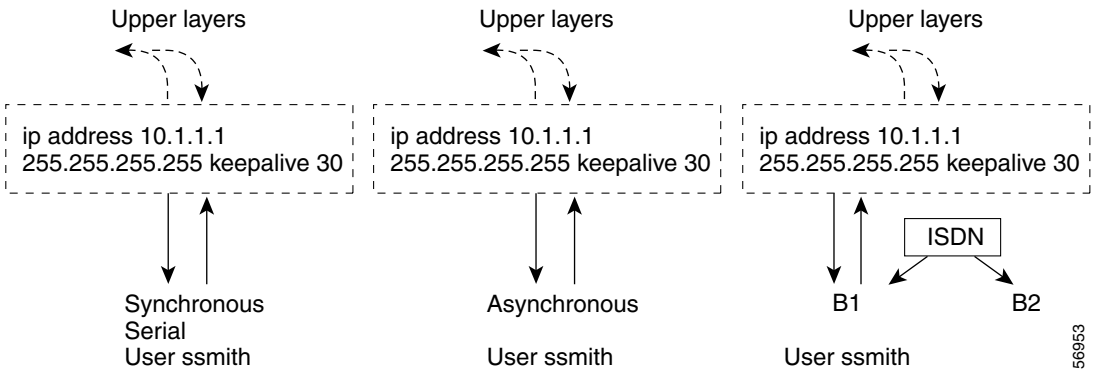
Suppose, for example, that the router interpreted the response by the AAA server as including only the following two commands for this user:

```
ip address 10.10.10.10 255.255.255.255
keepalive 30
```

In Figure 2, the dotted box represents the virtual profile configured only with the commands received from the AAA server, no matter which interface the incoming call arrived on. On the AAA RADIUS server, the attribute-value (AV) pair might have read as follows, where "\n" means to start a new command line:

```
cisco-avpair = "lcp:interface-config=ip address 10.10.10.10 255.255.255.0\nkeepalive 30",
```

Figure 2 Virtual Profiles by AAA Configuration



See the "Configuring Virtual Profiles by AAA Configuration" section on page 925 for configuration tasks for this case.

Case 3: Virtual Profiles Configured by Virtual Template and AAA Configuration

In this case, no DDR dialer profile is defined for the specific user, a virtual template for virtual profiles is defined, virtual profiles by AAA is enabled on the router, the router is configured for AAA, and a user-specific interface configuration for the user is stored on the AAA server.

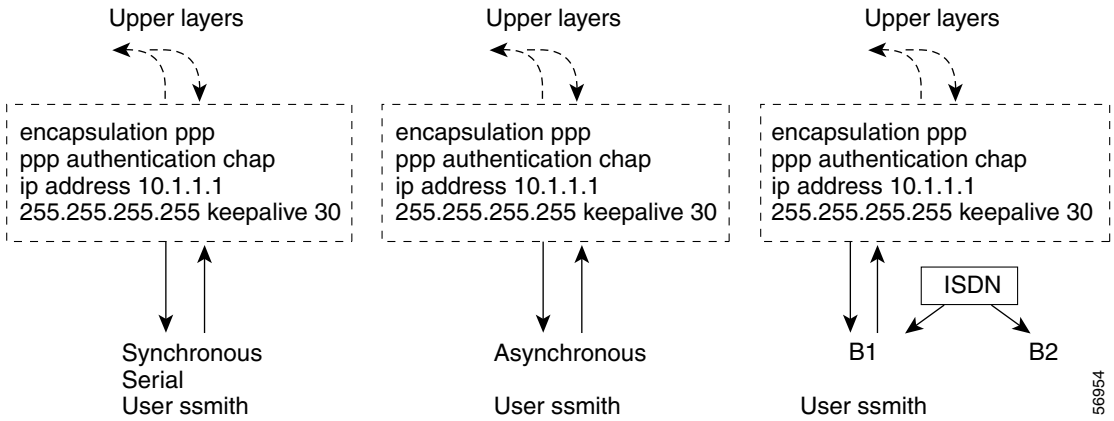
The router performs the following tasks in order:

1. Dynamically creates a virtual access interface cloned from the virtual template defined for virtual profiles.
2. Applies the user-specific interface configuration received from the AAA server.

If any command in the user's configuration conflicts with a command on the original interface or a command applied by cloning the virtual template, the user-specific command overrides the other command.

Suppose that the router had the virtual template as defined in Case 1 and the AAA user configuration as defined in Case 2. In Figure 3 the dotted box represents the virtual profile configured with configuration information from both sources, no matter which interface the incoming call arrived on. The **ip address** command has overridden the **ip unnumbered** command.

Figure 3 Virtual Profiles by Both Virtual Template and AAA Configuration



See the "Configuring Virtual Profiles by Both Virtual Template and AAA Configuration" section on page 925 for configuration tasks for this case.

Case 4: Virtual Profiles Configured by AAA, and a Virtual Template Defined by Another Application

In this case, no DDR dialer profile is defined for the specific user, virtual profiles by AAA are configured on the router but no virtual template is defined for virtual profiles, and a user-specific interface configuration is stored on the AAA server. In addition, a virtual template is configured for some other virtual access application (a VPDN, for example).

The router performs the following tasks in order:

1. Dynamically creates a virtual access interface and clones the virtual template from the other virtual access application onto it.
2. Applies the user-specific interface configuration received from the AAA server.

If any command in the virtual template conflicts with a command on the original interface, the template overrides it.

If any command in the AAA interface configuration for the user conflicts with a command in the virtual template, the user AAA interface configuration conflicts will override the virtual template.

If per-user configuration is also configured on the AAA server, that network protocol configuration is applied to the virtual access interface last.

The result is a virtual interface unique to that user.

How to Configure Virtual Profiles

To configure virtual profiles for dial-in users, perform the tasks in *one* of the first three sections and then troubleshoot the configuration by performing the tasks in the last section:

- Configuring Virtual Profiles by Virtual Template, page 924 (as required)
- Configuring Virtual Profiles by AAA Configuration, page 925 (as required)
- Configuring Virtual Profiles by Both Virtual Template and AAA Configuration, page 925 (as required)
- Troubleshooting Virtual Profile Configurations, page 927 (as required)

Note Do not define a DDR dialer profile for a user if you intend to define virtual profiles for the user.

Configuring Virtual Profiles by Virtual Template

To configure virtual profiles by virtual template, complete these two tasks:

- Creating and Configuring a Virtual Template Interface, page 924
- Specifying a Virtual Template Interface for Virtual Profiles, page 925

Note The order in which these tasks is performed is not crucial. However, both tasks must be completed before virtual profiles are used.

Creating and Configuring a Virtual Template Interface

Because a virtual template interface is a serial interface, all the configuration commands that apply to serial interfaces can also be applied to virtual template interfaces, except **shutdown** and **dialer** commands.

To create and configure a virtual template interface, use the following commands:

	Command	Purpose
Step 1	`Router(config)# interface virtual-template` *`number`*	Creates a virtual template interface and enters interface configuration mode.
Step 2	`Router(config-if)# ip unnumbered ethernet 0`	Enables IP without assigning a specific IP address on the LAN.
Step 3	`Router(config-if)# encapsulation ppp`	Enables PPP encapsulation on the virtual template interface.

Other optional PPP configuration commands can be added to the virtual template configuration. For example, you can add the **ppp authentication chap** command.

Specifying a Virtual Template Interface for Virtual Profiles

To specify a virtual template interface as the source of information for virtual profiles, use the following command:

Command	Purpose
`Router(config)# virtual-profile virtual-template` *number*	Specifies the virtual template interface as the source of information for virtual profiles.

Virtual template numbers range from 1 to 25.

Configuring Virtual Profiles by AAA Configuration

To configure virtual profiles by AAA only, complete these three tasks in any order. All tasks must be completed before virtual profiles are used.

- On the AAA server, create user-specific interface configurations for each of the specific users to use this method. See your AAA server documentation for more detailed configuration information about your AAA server.
- Configure AAA on the router, as described in the *Cisco IOS Security Configuration Guide*.
- Specify AAA as the source of information for virtual profiles.

To specify AAA as the source of information for virtual profiles, use the following command:

Command	Purpose
`Router(config)# virtual-profile aaa` **Note** Effective with Cisco IOS Release 12.2(34)SB and 12.2(33)XNE, the **virtual-profile aaa** command is not available in Cisco IOS software. In releases later than Cisco IOS Release 12.2, the router automatically creates virtual profiles when AAA attributes require a profile.	Specifies AAA as the source of user-specific interface configuration.

If you also want to use per-user configuration for network protocol access lists or route filters for individual users, see the chapter "Configuring Per-User Configuration" in this publication. In this case, no virtual template interface is defined for virtual profiles.

Configuring Virtual Profiles by Both Virtual Template and AAA Configuration

Use of user-specific AAA interface configuration information with virtual profiles requires the router to be configured for AAA and requires the AAA server to have user-specific interface configuration AV pairs. The relevant AV pairs (on a RADIUS server) begin as follows:

```
cisco-avpair = "lcp:interface-config=...",
```

The information that follows the equal sign (=) could be any Cisco IOS interface configuration command. For example, the line might be the following:

```
cisco-avpair = "lcp:interface-config=ip address 192.168.200.200 255.255.255.0",
```

Use of a virtual template interface with virtual profiles requires a virtual template to be defined specifically for virtual profiles.

To configure virtual profiles by both virtual template interface and AAA configuration, complete the following tasks in any order. All tasks must be completed before virtual profiles are used.

- On the AAA server, create user-specific interface configurations for each of the specific users to use this method. See your AAA server documentation for more detailed configuration information about your AAA server.
- Configure AAA on the router, as described in the *Cisco IOS Security Configuration Guide* publication.
- Creating and configuring a virtual template interface, described later in this chapter.
- Specifying virtual profiles by both virtual templates and AAA, described later in this chapter.

Creating and Configuring a Virtual Template Interface

To create and configure a virtual template interface, use the following commands:

	Command	Purpose
Step 1	`Router(config)# interface virtual-template` *number*	Creates a virtual template interface and enters interface configuration mode.
Step 2	`Router(config-if)# ip unnumbered ethernet 0`	Enables IP without assigning a specific IP address on the LAN.
Step 3	`Router(config-if)# encapsulation ppp`	Enables PPP encapsulation on the virtual template interface.

Because the software treats a virtual template interface as a serial interface, all the configuration commands that apply to serial interfaces can also be applied to virtual template interfaces, except **shutdown** and **dialer** commands. Other optional PPP configuration commands can also be added to the virtual template configuration. For example, you can add the **ppp authentication chap** command.

Specifying Virtual Profiles by Both Virtual Templates and AAA

To specify both the virtual template interface and the AAA per-user configuration as sources of information for virtual profiles, use the following commands:

	Command	Purpose
Step 1	`Router(config)# virtual-profile virtual-template` *number*	Defines the virtual template interface as the source of information for virtual profiles.
Step 2	`Router(config)# virtual-profile aaa`	Specifies AAA as the source of user-specific configuration for virtual profiles.

If you also want to use per-user configuration for network protocol access lists or route filters for individual users, see the Configuring per-User Configuration feature.

Troubleshooting Virtual Profile Configurations

To troubleshoot the virtual profiles configurations, use any of the following **debug** commands:

Command	Purpose
`Router# debug dialer`	Displays information about dial calls and negotiations and virtual profile events.
`Router# debug aaa per-user`	Displays information about the per-user configuration downloaded from the AAA server.
`Router# debug vtemplate cloning`	Displays cloning information for a virtual access interface from the time it is cloned from a virtual template to the time it comes down.

Configuration Examples for Virtual Profiles

The following sections provide examples for the four cases described in this chapter:

- Virtual Profiles Configured by Virtual Templates: Example, page 927
- Virtual Profiles Configured by AAA Configuration: Example, page 929
- Virtual Profiles Configured by Virtual Templates and AAA Configuration: Example, page 930
- Virtual Profiles Configured by AAA Plus a VPDN Virtual Template on a VPDN Home Gateway: Example, page 931

In these examples, BRI 0 is configured for legacy DDR, and interface BRI 1 is configured for dialer profiles. Note that interface dialer 0 is configured for legacy DDR. Interface dialer 1 is a dialer profile.

The intention of the examples is to show how to configure virtual profiles. In addition, the examples show the interoperability of DDR and dialer profiles in the respective cases with various forms of virtual profiles.

The same user names (User1 and User2) occur in all these examples. Note the different configuration allowed to them in each of the four examples.

User1 is a normal user and can dial in to BRI 0 only. User2 is a privileged user who can dial in to BRI 0 and BRI 1. If User2 dials into BRI 1, the dialer profile will be used. If User2 dials into BRI 0, virtual profiles will be used. Because User1 does not have a dialer profile, only virtual profiles can be applied to User1.

To see an example of a configuration using virtual profiles and the Dynamic Multiple Encapsulations feature, see the “Multiple Encapsulations over ISDN” example in the chapter “Configuring Peer-to-Peer DDR with Dialer Profiles.”

Virtual Profiles Configured by Virtual Templates: Example

The following example shows a router configured for virtual profiles by virtual template. (Virtual profiles do not have any interface-specific AAA configuration.) Comments in the example draw attention to specific features or ignored lines.

In this example, the same virtual template interface applies to both users; they have the same interface configurations.

Router Configuration

```
! Enable AAA on the router.
aaa new-model
aaa authentication ppp default radius
! The following command is required.
aaa authorization network radius
enable secret 5 $1$koOn$/1QAylov6JFAElxRCrL.o/
enable password lab
!
! Specify configuration of virtual profiles by virtual template.
! This is the key command for this example.
virtual-profile virtual-template 1
!
! Define the virtual template.
interface Virtual-Template 1
 ip unnumbered ethernet 0
 encapsulation ppp
 ppp authentication chap
!
switch-type basic-dms100
interface BRI 0
 description Connected to 103
 encapsulation ppp
 no ip route-cache
 dialer rotary-group 0
 ppp authentication chap
!
interface BRI 1
 description Connected to 104
 encapsulation ppp
! Disable fast switching.
 no ip route-cache
 dialer pool-member 1
 ppp authentication chap
!
! Configure dialer interface 0 for DDR for User1 and User2.
interface dialer 0
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
! Enable legacy DDR.
 dialer in-band
! Disable fast switching.
 no ip route-cache
 dialer map ip 10.1.1.2 name User1 1111
 dialer map ip 10.1.1.3 name User2 2222
 dialer-group 1
 ppp authentication chap
!
! Configure dialer interface 1 for DDR to dial out to User2.
interface dialer 1
 ip address 10.2.2.2 255.255.255.0
 encapsulation ppp
 dialer remote-name User2
 dialer string 3333
 dialer pool 1
 dialer-group 1
! Disable fast switching.
 no ip route-cache
 ppp authentication chap
 dialer-list 1 protocol ip permit
```

Virtual Profiles Configured by AAA Configuration: Example

The following example shows the router configuration for virtual profiles by AAA and the AAA server configuration for user-specific interface configurations. User1 and User2 have different IP addresses.

In the AAA configuration cisco-avpair lines, "\n" is used to indicate the start of a new Cisco IOS command line.

AAA Configuration for User1 and User2

```
User1 Password = "welcome"
 User-Service-Type = Framed-User,
 Framed-Protocol = PPP,
 cisco-avpair = "lcp:interface-config=keepalive 75\nip address 192.16.100.100
 255.255.255.0",
User2 Password = "emoclew"
 User-Service-Type = Framed-User,
 Framed-Protocol = PPP,
 cisco-avpair = "lcp:interface-config=keepalive 100\nip address 192.168.200.200
 255.255.255.0"
```

Router Configuration

```
! Enable AAA on the router.
aaa new-model
aaa authentication ppp default radius
! This is a key command for this example.
aaa authorization network radius
enable secret 5 $1$koOn$/1QAylov6JFAElxRCrL.o/
enable password lab
!
! Specify configuration of virtual profiles by aaa.
! This is a key command for this example.
virtual-profiles aaa
!
! Interface BRI 0 is configured for legacy DDR.
interface BRI 0
 description Connected to 103
 encapsulation ppp
 no ip route-cache
 dialer rotary-group 0
 ppp authentication chap
!
! Interface BRI 1 is configured for dialer profiles.
interface BRI 1
 description Connected to 104
 encapsulation ppp
! Disable fast switching.
 no ip route-cache
 dialer pool-member 1
 ppp authentication chap
!
! Configure dialer interface 0 for DDR for User1 and User2.
interface dialer 0
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
! Enable legacy DDR.
 dialer in-band
! Disable fast switching.
 no ip route-cache
 dialer map ip 10.1.1.2 name User1 1111
 dialer map ip 10.1.1.3 name User2 2222
 dialer-group 1
```

```
 ppp authentication chap
!
! Configure dialer interface 1 for DDR to dial out to User2.
interface dialer 1
 ip address 10.2.2.2 255.255.255.0
 encapsulation ppp
 dialer remote-name User2
 dialer string 3333
 dialer pool 1
 dialer-group 1
! Disable fast switching.
 no ip route-cache
 ppp authentication chap
 dialer-list 1 protocol ip permit
```

Virtual Profiles Configured by Virtual Templates and AAA Configuration: Example

The following example shows how virtual profiles can be configured by both virtual templates and AAA configuration. User1 and User2 can dial in from anywhere and have their same keepalive settings and their own IP addresses.

The remaining AV pair settings are not used by virtual profiles. They are the network protocol access lists and route filters used by AAA-based per-user configuration.

In the AAA configuration cisco-avpair lines, "\n" is used to indicate the start of a new Cisco IOS command line.

AAA Configuration for User1 and User2

```
User1 Password = "welcome"
     User-Service-Type = Framed-User,
     Framed-Protocol = PPP,
        cisco-avpair = "lcp:interface-config=keepalive 75\nip address 10.16.100.100
255.255.255.0",
        cisco-avpair = "ip:rte-fltr-out#0=router igrp 60",
        cisco-avpair = "ip:rte-fltr-out#3=deny 172.16.0.0 0.255.255.255",
        cisco-avpair = "ip:rte-fltr-out#4=deny 172.17.0.0 0.255.255.255",
        cisco-avpair = "ip:rte-fltr-out#5=permit any"
User2 Password = "emoclew"
     User-Service-Type = Framed-User,
     Framed-Protocol = PPP,
        cisco-avpair = "lcp:interface-config=keepalive 100\nip address 192.168.200.200
255.255.255.0",
        cisco-avpair = "ip:inacl#3=permit ip any any precedence immediate",
        cisco-avpair = "ip:inacl#4=deny igrp 10.0.1.2 255.255.0.0 any",
        cisco-avpair = "ip:outacl#2=permit ip any any precedence immediate",
        cisco-avpair = "ip:outacl#3=deny igrp 10.0.9.10 255.255.0.0 any"
```

Router Configuration

```
! Enable AAA on the router.
aaa new-model
aaa authentication ppp default radius
! This is a key command for this example.
aaa authorization network radius
enable secret 5 $1$koOn$/1QAylov6JFAElxRCrL.o/
enable password lab
!
! Specify use of virtual profiles and a virtual template.
! The following two commands are key for this example.
```

```
virtual-profile virtual-template 1
virtual-profile aaa
!
! Define the virtual template.
interface Virtual-Template 1
 ip unnumbered ethernet 0
 encapsulation ppp
 ppp authentication chap
!
! Interface BRI 0 is configured for legacy DDR.
interface BRI 0
 description Connected to 103
 encapsulation ppp
 no ip route-cache
 dialer rotary-group 0
 ppp authentication chap
!
! Interface BRI 1 is configured for dialer profiles.
interface BRI 1
 description Connected to 104
 encapsulation ppp
! Disable fast switching.
 no ip route-cache
 dialer pool-member 1
 ppp authentication chap
!
! Configure dialer interface 0 for DDR to dial out to User1 and User2.
interface dialer 0
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
 dialer in-band
! Disable fast switching.
 no ip route-cache
 dialer map ip 10.1.1.2 name User1 1111
 dialer map ip 10.1.1.3 name User2 2222
 dialer-group 1
 ppp authentication chap
!
! Configure dialer interface 0 for DDR to dial out to User2.
interface dialer 1
 ip address 10.2.2.2 255.255.255.0
 encapsulation ppp
 dialer remote-name User2
 dialer string 3333
 dialer pool 1
 dialer-group 1
! Disable fast switching.
 no ip route-cache
 ppp authentication chap
!
 dialer-list 1 protocol ip permit
```

Virtual Profiles Configured by AAA Plus a VPDN Virtual Template on a VPDN Home Gateway: Example

Like the virtual profiles configured by AAA example earlier in this section, the following example shows the router configuration for virtual profiles by AAA. The user file on the AAA server also includes interface configuration for User1 and User2, the two users. Specifically, User1 and User2 each have their own IP addresses when they are in privileged mode.

In this case, however, the router is also configured as the VPDN home gateway. It clones the VPDN virtual template interface first and then clones the virtual profiles AAA interface configuration. If per-user configuration were configured on this router and the user file on the AAA server had network protocol information for the two users, that information would be applied to the virtual access interface last.

In the AAA configuration cisco-avpair lines, "\n" is used to indicate the start of a new Cisco IOS command line.

AAA Configuration for User1 and User2

```
User1 Password = "welcome"
 User-Service-Type = Framed-User,
 Framed-Protocol = PPP,
 cisco-avpair = "lcp:interface-config=keepalive 75\nip address 10.100.100.100
 255.255.255.0",
User2 Password = "emoclew"
 User-Service-Type = Framed-User,
 Framed-Protocol = PPP,
 cisco-avpair = "lcp:interface-config=keepalive 100\nip address 192.168.200.200
 255.255.255.0"
```

Router Configuration

```
!Configure the router as the VPDN home gateway.
!
!Enable VPDN and specify the VPDN virtual template to use on incoming calls from the
!network access server.
vpdn enable
vpdn incoming dallas_wan go_blue virtual-template 6
!
!Configure the virtual template interface for VPDN.
interface virtual template 6
ip unnumbered ethernet 0
encapsulation ppp
ppp authentication chap
!
!Enable AAA on the router.
aaa new-model
aaa authentication ppp default radius
aaa authorization network radius
enable secret 5 $1$koOn$/1QAylov6JFAElxRCrL.o/
enable password lab
!
!Specify configuration of virtual profiles by aaa.
virtual-profiles aaa
!
!Configure the physical synchronous serial 0 interface.
interface Serial 0
 description Connected to 101
 encapsulation ppp
!Disable fast switching.
 no ip route-cache
 ppp authentication chap
!
!Configure serial interface 1 for DDR. S1 uses dialer rotary group 0, which is
!defined on BRI interface 0.
interface serial 1
 description Connected to 102
 encapsulation ppp
 dialer in-band
! Disable fast switching.
 no ip route-cache
```

```
 dialer rotary-group 0
 ppp authentication chap
!
interface BRI 0
 description Connected to 103
 encapsulation ppp
 no ip route-cache
 dialer rotary-group 0
 ppp authentication chap
!
interface BRI 1
 description Connected to 104
 encapsulation ppp
!Disable fast switching.
 no ip route-cache
 dialer pool-member 1
 ppp authentication chap
!
!Configure dialer interface 0 for DDR to call and receive calls from User1 and User2.
interface dialer 0
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
!Enable legacy DDR.
 dialer in-band
!Disable fast switching.
 no ip route-cache
 dialer map ip 10.1.1.2 name User1 1111
 dialer map ip 10.1.1.3 name User2 2222
 dialer-group 1
 ppp authentication chap
!
!Configure dialer interface 1 for DDR to dial out to User2.
interface dialer 1
 ip address 10.2.2.2 255.255.255.0
 encapsulation ppp
 dialer remote-name User2
 dialer string 3333
 dialer pool 1
 dialer-group 1
!Disable fast switching.
 no ip route-cache
 ppp authentication chap
 dialer-list 1 protocol ip permit
```

Additional References

The following sections provide references related to configuring virtual profiles.

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
Dial commands	*Cisco IOS Dial Command Reference*

Standards

Standard	Title
None	—

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
None	—

Technical Assistance

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

Feature Information for Configuring Virtual Profiles

Table 2 lists the features in this module and provides links to specific configuration information.

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

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 2 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 2 Feature Information for Configuring Virtual Profiles

Feature Name	Releases	Feature Information
Configuring Virtual Profiles	11.3	A virtual profile is a unique application that can create and configure a virtual access interface dynamically when a dial-in call is received and that can tear down the interface dynamically when the call ends.

IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

First Published: March 16, 2012
Last Updated: July 30, 2014

The IPv6 Cisco Express Forwarding Switching on Dialer Interfaces feature allows Cisco Express Forwarding switching of IPv6 traffic on dialer interfaces.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for IPv6 Cisco Express Forwarding Switching on Dialer Interfaces" section on page 9.

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

Restrictions for IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

- IPv6 Cisco Express Forwarding switching is supported only on dialer-profile configuration. IPv6 Cisco Express Forwarding switching is not supported on dialer-legacy and dialer-rotary configurations because there is no mechanism for peer global prefix negotiation. For dialer-legacy and dialer-rotary configurations, the IPv6 packets are process switched.
- In case of Point-to-Point Protocol over Ethernet (PPPoE) configurations with IPv6 enabled, counters have to be updated on virtual-access interface for ipIfStatsOutOctets OID.

Information About IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

Dialer Watch

Dialer watch is a method of implementing redundancy or a backup system in case of a router failure. By configuring a set of watched routes that define the primary interface, you can monitor the status of the primary interfaces while watched routes are added or deleted.

The monitoring is performed in the following sequence:

1. Whenever a watched route is deleted, the dialer watch mechanism checks for a valid route for any of the defined watched IP addresses.
2. If no valid route exists, the primary link is considered down and unusable.
3. If a valid route exists for at least one of the defined IP addresses, and if the route is pointing to an interface other than the backup interface configured for dialer watch, the primary link is considered as up.
4. If the primary link goes down, the dialer watch is immediately notified by the routing protocol and the secondary link is brought up.
5. When the secondary link is up, after the idle timeout expires, the status of the primary link is rechecked.
6. If the primary link remains down, the idle timer is reset.
7. If the primary link is up, the secondary backup link is disconnected. By using the **dialer watch-list delay** command, you can create a delay for the secondary link to be disconnected after the primary link is reestablished.

IPv6 Cisco Express Forwarding

Cisco Express Forwarding is used to switch IPv6 packets on dialer interfaces. IPv6 packets are Cisco Express Forwarding-switched in inbound and outbound traffic, irrespective of the dialer mode.

Cisco Express Forwarding support for inbound traffic on dialer interfaces requires the virtual-access interfaces supporting the dialer to be IPv6-enabled and available in the same VPN routing and forwarding (VRF) instance as the dialer interface.

With CSCtk62149, the IPv6 Cisco Express Forwarding Switching on Dialer Interfaces feature is supported. For information about the Cisco Express Forwarding of IPv4 Traffic on Dialer Interfaces feature, see the “Configuring Dialer Cisco Express Forwarding” module in the *Dial Technologies Configuration Guide*.

Cisco Express Forwarding with IPv4 and IPv6 Packets

When both IPv4 and IPv6 are configured on a peer, the IP Control Protocol (IPCP) negotiates with a peer IPv4 address, and the IPv6CP negotiates with the interface and forms the peer link local address. When the PPP negotiation is successful, the IPv4 and IPv6 adjacency is complete on the dialer, thereby enabling Cisco Express Forwarding switching of IPv4 and IPv6 packets to the peer.

How to Configure IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

Configuring Cisco Express Forwarding Switching of IPv6 Traffic

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** *ipv6-address-prefix*
5. **ipv6 enable**
6. **exit**
7. **ipv6 unicast-routing**
8. **ipv6 cef**
9. **dialer watch-list** *group-number* **ipv6** *ipv6-address ipv6-address-mask* [**vrf** *vrf-name*]
10. Repeat Step 9 to define watch-list for each IPv6 address or IPv6-VRF pair to be monitored.
11. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface dialer 2`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-address-prefix* **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64`	Assigns an IPv6 address to the interface.
Step 5	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on the interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 8	**ipv6 cef** **Example:** `Device(config)# ipv6 cef`	Enables Cisco Express Forwarding globally on the device.
Step 9	**dialer watch-list** *group-number* **ipv6** *ipv6-address ipv6-address-mask* [**vrf** *vrf-name*] **Example:** `Device(config)# dialer watch-list 4 ipv6 2001:DB8:0:ABCD::1 FFFF:FFFF::`	Defines the IPv6 adress route to be watched. • If the VRF instance is specified, this command defines the IPv6 address and the VRF instance pair to be monitored.
Step 10	Repeat Step 9 to define watch-list for each IPv6 address or IPv6-VRF pair to be monitored.	—
Step 11	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

Example: Configuring the Cisco Express Forwarding Switching of IPv6 Traffic

```
Device# configure terminal
Device(config)# interface dialer 2
Device(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64
Device(config-if)# ipv6 enable
Device(config-if)# exit
Device(config)# ipv6 unicast-routing
Device(config)# ipv6 cef
Device(config)# dialer watch-list 4 ipv6 2001:DB8:0:ABCD::1 FFFF:FFFF::
Device(config)# dialer watch-list 4 ipv6 2001:DB8:0:ABCD::8 FFFF:FFFF:: vrf vrf4
Device(config)# exit
```

Example: Configuring the Dialer Watch to Monitor IPv6 Addresses

```
Device# configure terminal
Device(config)# dialer watch-list 1 ipv6 2001:DB8:0:ABCD::1 FFFF:FFFF::
Device(config)# dialer watch-list 1 ipv6 2001:DB8:0:ABCD::2 FFFF:FFFF::
Device(config)# dialer watch-list 1 ipv6 2001:DB8:0:ABCD::3 FFFF:FFFF::
Device(config)# exit
```

Example: Configuring the Dialer Watch to Monitor IPv6 Addresses and VRF Pairs

```
Device# configure terminal
Device(config)# dialer watch-list 2 ipv6 2001:DB8:0:ABCD::1 FFFF:FFFF::
Device(config)# dialer watch-list 2 ipv6 2001:DB8:0:ABCD::2 FFFF:FFFF:: vrf vrf1
Device(config)# dialer watch-list 2 ipv6 2001:DB8:0:ABCD::3 FFFF:FFFF:: vrf vrf2
Device(config)# exit
```

Example: Sample Configuration of IPv4 and IPv6 Cisco Express Forwarding on Dialer Interfaces

```
Device# show running-config interface dialer 1

Building configuration...

Current configuration : 1473 bytes
!
! Last configuration change at 08:53:54 IST Tue Feb 14 2012
version 15.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname r601
!
boot-start-marker
```

```
boot-end-marker
!
no aaa new-model
clock timezone IST 0 0
mmi polling-interval 60
no mmi auto-configure
no mmi pvc
mmi snmp-timeout 180
!
ip cef
ipv6 unicast-routing
ipv6 cef
multilink bundle-name authenticated
isdn switch-type primary-5ess
!
crypto pki token default removal timeout 0
!
controller T1 7/0
pri-group timeslots 1-24
!
interface Ethernet0/0
ip address 10.1.1.2 255.255.255.0
ipv6 address 2001:DB8:1::1122/64
!
interface Ethernet0/1
no ip address
shutdown
!
interface Ethernet0/2
no ip address
shutdown
!
interface Ethernet0/3
no ip address
shutdown
!
interface Serial7/0:23
no ip address
encapsulation ppp
dialer rotary-group 1
dialer-group 1
isdn switch-type primary-5ess
isdn protocol-emulate network
!
interface Async1
no ip address
encapsulation slip
!
interface Dialer1
ip address 10.2.2.2 255.255.255.0
encapsulation ppp
dialer in-band
dialer idle-timeout 60
dialer string 1234
dialer-group 1
ipv6 address 2001:DB8:2::1234/64
!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
dialer-list 1 protocol ipv6 permit
```

```
dialer-list 1 protocol ip permit
!
line con 0
logging synchronous
line aux 0
line vty 0 4
login
transport input all
!
end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Master Commands List, All Releases*
Dial Technology commands	*Dial Technologies Command Reference*
Cisco Express Forwarding	• "Cisco IOS Switching Paths Overview" module in the *Switching Services Configuration Guide* • *IP Switching Command Reference*
Dialer interfaces and profiles; Dialer Cisco Express forwarding	• "Configuring Peer-to-Peer DDR with Dialer Profiles" module in the *Dial Technologies Configuration Guide* • "Configuring Dialer Cisco Express Forwarding" module in the *Dial Technologies Configuration Guide*

MIBs

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

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

Table 1 lists the release history for this feature.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 1 lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 1 Feature Information for IPv6 Cisco Express Forwarding Switching on Dialer Interfaces

Feature Name	Releases	Feature Information
IPv6 Cisco Express Forwarding Switching on Dialer Interfaces	15.2(3)T	The IPv6 Cisco Express Forwarding Switching on Dialer Interfaces feature allows Cisco Express Forwarding switching of IPv6 traffic on dialer interfaces. The following command was introduced or modified: **dialer watch-list**.
IPv6 Cisco Express Forwarding Switching on Dialer Interfaces	Cisco IOS XE Release 3.9S	The IPv6 Cisco Express Forwarding Switching on Dialer Interfaces feature allows Cisco Express Forwarding switching of IPv6 traffic on dialer interfaces. The following command was introduced or modified: **dialer watch-list**.

Configuring Asynchronous SLIP and PPP

First Published: November 4, 1996
Last Updated: November 20, 2014

The Configuring Asynchronous SLIP and PPP module describes how to configure asynchronous Serial Line Internet Protocol (SLIP) and PPP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Configuring Asynchronous SLIP and PPP" section on page 960.

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

Contents

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **command** [**keyword** *argument*]
4. **command** [**keyword** *argument*]

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		
Step 4		

Examples

<<Text.>>

Troubleshooting Tips

<<Text.>>

What to Do Next

<<Text.>>

Configuration Examples for <Phrase Based on Module Title>

<<Text.>>

- <<Text.>>

Example: <xxx>

<<Text.>>

Information About Configuring Asynchronous SLIP and PPP

Asynchronous SLIP and PPP Overview

PPP and SLIP define methods of sending IP packets over standard asynchronous serial lines with minimum line speeds of 1200 baud.

Using SLIP or PPP encapsulation over asynchronous lines is an inexpensive way to connect personal computers (PCs) to a network. PPP and SLIP over asynchronous dialup modems allow a home computer to be connected to a network without the cost of a leased line. Dialup PPP and SLIP links can also be used for remote sites that need only occasional remote node or backup connectivity. Both public-domain and vendor-supported PPP and SLIP implementations are available for a variety of computer applications.

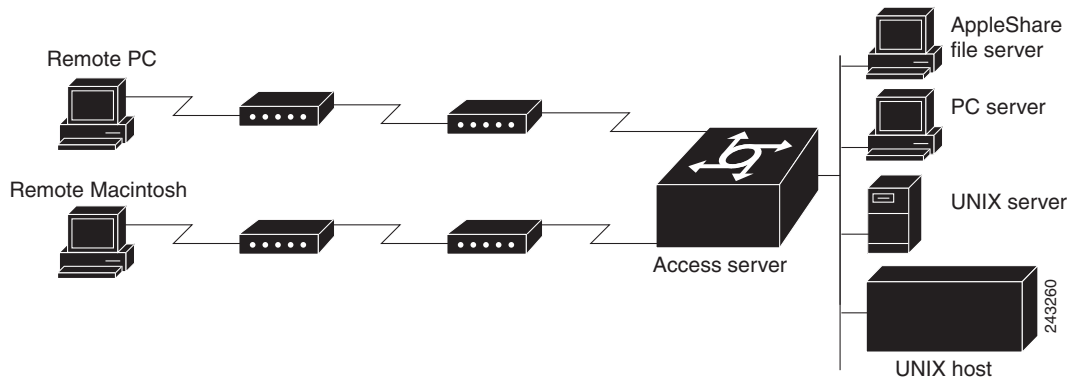
The Cisco IOS software concentrates a large number of SLIP or PPP PC or workstation client hosts onto a network interface that allows the PCs to communicate with any host on the network. The Cisco IOS software can support any combination of SLIP or PPP lines and lines dedicated to normal asynchronous devices such as terminals and modems.

SLIP is an older protocol. PPP is a newer, more robust protocol than SLIP, and it contains functions that can detect or prevent misconfiguration. PPP also provides greater built-in security mechanisms.

Note Most asynchronous serial links have very low bandwidth. Take care to configure your system so the links will not be overloaded. Consider using default routes and filtering routing updates to prevent them from being sent on these asynchronous lines.

Figure 1 illustrates a typical asynchronous SLIP or PPP remote-node configuration.

Figure 1 Sample SLIP or PPP Remote-Node Configuration



Responding to BOOTP Requests

The Bootstrap (BOOTP) protocol allows a client machine to discover its own IP address, the address of the router, and the name of a file to be loaded in to memory and executed. There are typically two phases to using BOOTP: first, the client's address is determined and the boot file is selected; then the file is transferred, using the TFTP.

PPP and SLIP clients can send BOOTP requests to the Cisco IOS software, and the Cisco IOS software responds with information about the network. For example, the client can send a BOOTP request to learn its IP address and where the boot file is located, and the Cisco IOS software responds with the information.

BOOTP supports the extended BOOTP requests specified in RFC 1084 and works for both PPP and SLIP encapsulation.

BOOTP compares to Reverse Address Resolution Protocol (RARP) as follows: RARP is an older protocol that allows a client to determine its IP address if it knows its hardware address. (Refer to the *IP Configuration Guide* for more information about RARP.) However, RARP is a hardware link protocol, and can be implemented only on hosts that have special kernel or driver modifications that allow access to these raw packets. BOOTP does not require kernel modifications.

Asynchronous Network Connections and Routing

Line configuration commands configure a connection to a terminal or a modem. Interface configuration (**async**) commands, described in this chapter, configure a line as an asynchronous network interface over which networking functions are performed.

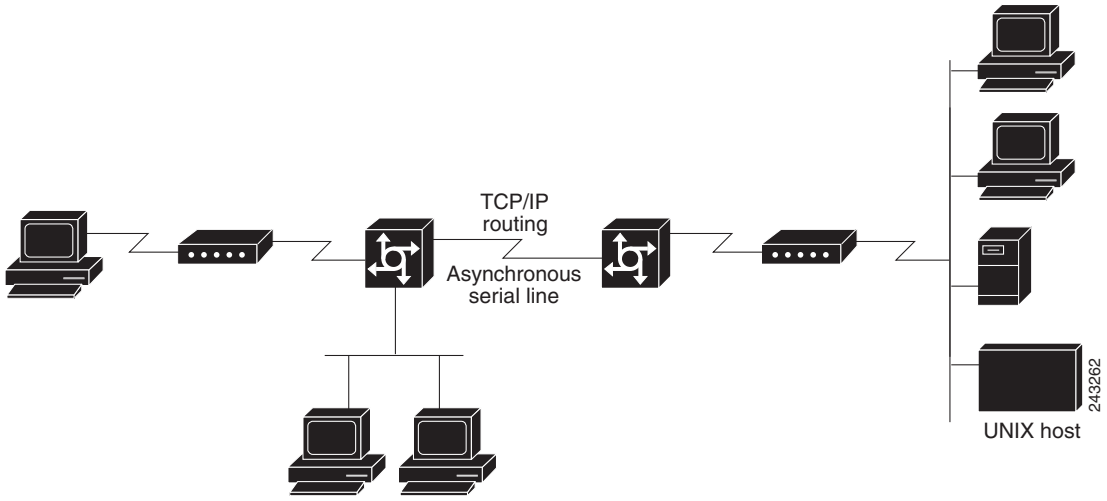
The Cisco IOS software also supports IP routing connections for communication that requires connecting one network to another.

The Cisco IOS software supports protocol translation for PPP and SLIP between other network devices running Telnet, local-area transport (LAT), or X.25. For example, you can send IP packets across a public X.25 packet assembler/disassembler (PAD) network using SLIP or PPP encapsulation when SLIP or PPP protocol translation is enabled. For more information, see the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" in *Dial Technologies Configuration Guide*.

If asynchronous dynamic routing is enabled, you can enable routing at the user level by using the **routing** keyword with the **slip** or **ppp** commands.

Asynchronous interfaces offer both dedicated and dynamic address assignment, configurable hold queues and IP packet sizes, extended BOOTP requests, and permit and deny conditions for controlling access to lines. Figure 2 shows a sample asynchronous routing configuration.

Figure 2 Sample Asynchronous Routing Configuration



Asynchronous Interfaces and Broadcasts

The Cisco IOS software recognizes a variety of IP broadcast addresses. When a router receives an IP packet from an asynchronous client, it rebroadcasts the packet onto the network without changing the IP header.

The Cisco IOS software receives the SLIP or PPP client broadcasts and responds to BOOTP requests with the current IP address assigned to the asynchronous interface from which the request was received. This facility allows the asynchronous client software to automatically learn its own IP address.

Network-Layer Protocols over PPP and SLIP

You can configure network-layer protocols, such as AppleTalk, IP, and Internet Protocol Exchange (IPX), over PPP and SLIP. SLIP supports only IP, but PPP supports each of these protocols.

Configuring IPX and PPP

You can configure IPX over PPP (IPXCP) on synchronous serial and asynchronous serial interfaces using one of two methods.

The first method associates an asynchronous interface with a loopback interface configured to run IPX. It permits you to configure IPX-PPP on asynchronous interfaces only.

The second method permits you to configure IPX-PPP on asynchronous and synchronous serial interfaces. However, it requires that you specify a dedicated IPX network number for each interface, which can require a substantial number of network numbers for a large number of interfaces.

You can also configure IPX to run on virtual terminal lines configured for PPP. See the section “Enabling IPX and PPP over X.25 to an IPX Network on Virtual Terminal Lines” in this chapter.

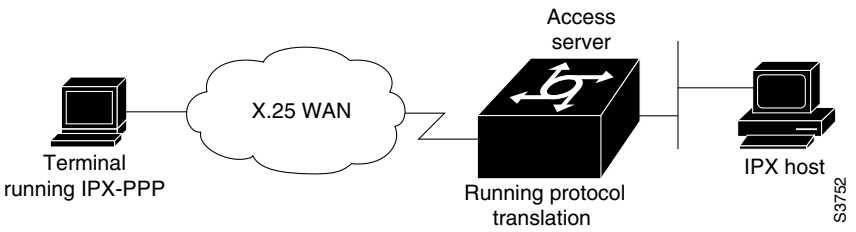
Note If you are configuring IPX-PPP on asynchronous interfaces, you should filter routing updates on the interface. Most asynchronous serial links have very low bandwidth, and routing updates take up a great deal of bandwidth. The previous task table uses the **ipx update interval** command to filter SAP updates. For more information about filtering routing updates, see the section about creating filters for updating the routing table in the chapter "Configuring Novell IPX" in the *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

IPX and PPP over X.25 to an IPX Network on Virtual Terminal Lines

You can enable IPX-PPP on virtual terminal lines, which permits clients to log in to a virtual terminal on a router, invoke a PPP session at the EXEC prompt to a host, and run IPX to the host.

For example, in Figure 3, the client terminal on the X.25 network logs in to the access server via a virtual terminal line, which is configured for IPX-PPP. When the user connects to the access server and the EXEC prompt appears, enter the PPP command to connect to the IPX host. The virtual terminal is configured to run IPX, so when the PPP session is established from the access server, the terminal can access the IPX host using an IPX application.

Figure 3 IPX-PPP on a Virtual Asynchronous Interface



AppleTalk and PPP

You can configure an asynchronous interface so that users can access AppleTalk zones by dialing in to the router via PPP through this interface. Users accessing the network can run AppleTalk and IP natively on a remote Macintosh, access any available AppleTalk zones from Chooser, use networked peripherals, and share files with other Macintosh users. This feature is referred to as AppleTalk Control Protocol (ATCP).

You create a virtual network that exists only for accessing an AppleTalk internet through the server. To create a new AppleTalk zone, enter the **appletalk virtual-net** command and use a new zone name; this network number is then the only one associated with this zone. To add network numbers to an existing AppleTalk zone, use this existing zone name in the command; this network number is then added to the existing zone. Routing is not supported on these interfaces.

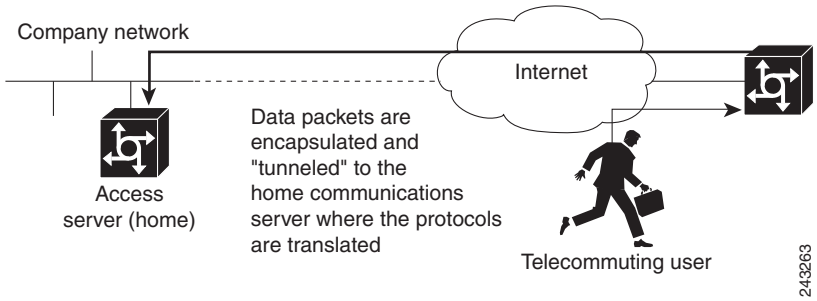
Asynchronous Host Mobility

The access server supports a packet tunneling strategy that extends the internetwork—in effect creating a virtual private link for the mobile user. When a user activates asynchronous host mobility, the access server on which the remote user dials in becomes a remote point of presence (POP) for the home network of the user. Once logged in, users experience a server environment identical to the one that they experience when they connect directly to the "home" access server.

Once the network-layer connection is made, data packets are tunneled at the physical or data link layer instead of at the protocol layer. In this way, raw data bytes from dial-in users are transported directly to the "home" access server, which processes the protocols.

Figure 4 illustrates the implementation of asynchronous host mobility on an extended internetwork. A mobile user connects to an access server on the internetwork and, by activating asynchronous host mobility, is connected to a "home" access server configured with the appropriate username. The user sees an authentication dialog or prompt from the "home" system and can proceed as if he or she were connected directly to that device.

Figure 4 Asynchronous Host Mobility



Asynchronous host mobility is enabled with the **tunnel** EXEC command and the **ip tcp async-mobility server** global configuration command. The **ip tcp async-mobility server** command establishes asynchronous listening on TCP tunnel port 57. The **tunnel** command sets up a network-layer connection to the specified destination. Both commands must be used. The access server accepts the connection, attaches it to a virtual terminal line, and runs a command parser capable of running the normal dial-in services. After the connection is established, data is transferred between the modem and network connection with a minimum of interpretations. When communications are complete, the network connection can be closed and terminated from either end.

To connect from a router other than a Cisco router, you must use Telnet. After a connection is established, you receive an authentication dialog or prompt from your home router, and can proceed as if you are connected directly to that router. When communications are complete, the network connection can be closed and terminated from either end of the connection.

Additional Remote Node Connections

This section describes how to connect devices across telephone lines by using PPP and SLIP. It includes the following sections:

- Creating PPP Connections
- Making SLIP Connections

Creating PPP Connections

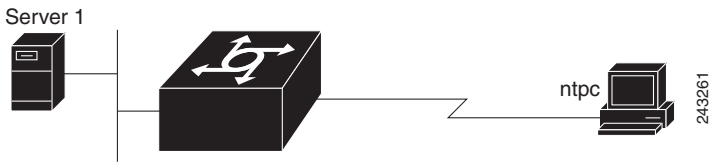
When you connect from a remote node computer through an asynchronous port on an access server to the EXEC facility to connect from the access server to a device on the network, use the following command in EXEC mode:

Command	Purpose
Router> **ppp** {**/default** | {*remote-ip-address* | *remote-name*} [**@***tacacs-server*]} [**/routing**]	Creates a PPP connection.

If you specify an address for the TACACS server using **/default** or *tacacs-server*, the address must be the first parameter in the command after you type **ppp**. If you do not specify an address or enter **/default**, you are prompted for an IP address or host name. You can enter **/default** at this point.

For example, if you are working at home on the device named ntpc in Figure 5 and want to connect to Server 1 using PPP, you could dial in to the access server. When you connect to the EXEC prompt on the access server, enter the **ppp** command to connect with the device.

Figure 5 Using the ppp Command



To terminate a session, disconnect from the device on the network using the command specific to that device. Then, exit from EXEC mode by using the **exit** command.

Making SLIP Connections

To make a serial connection to a remote host by using SLIP, use the following command in EXEC mode:

Command	Purpose
Router> **slip** [**/default**] {*remote-ip-address* | *remote-name*} [**@***tacacs-server*] [**/routing**]} [**/compressed**]	Creates a SLIP connection.

Your system administrator can configure SLIP to expect a specific address or to provide one for you. It is also possible to set up SLIP in a mode that compresses packets for more efficient use of bandwidth on the line.

If you specify an address for the TACACS server using **/default** or *tacacs-server*, the address must be the first parameter in the command after you type **slip**. If you do not specify an address or enter **/default**, you are prompted for an IP address or host name. You can enter **/default** at this point.

If you do not use the *tacacs-server* argument to specify a TACACS server for SLIP address authentication, the TACACS server specified at login (if any) is used for the SLIP address query.

To optimize bandwidth on a line, SLIP enables compression of the SLIP packets using Van Jacobson TCP header compression as defined in RFC 1144.

To terminate a session, disconnect from the device on the network using the command specific to that device. Then, exit from EXEC mode by using the **exit** command.

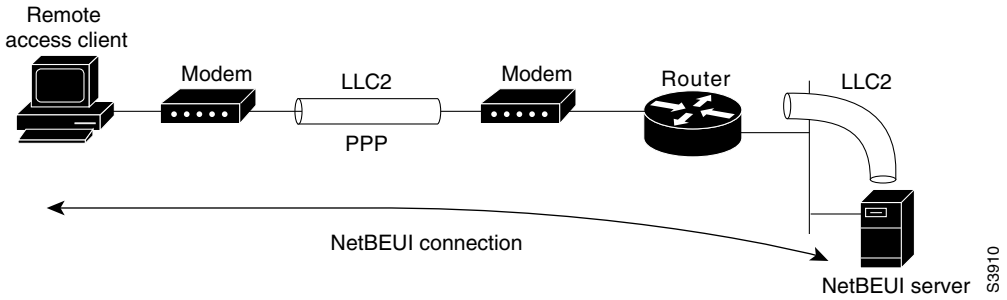
Remote Access to NetBEUI Services

NetBIOS Extended User Interface (NetBEUI) is a simple networking protocol developed by IBM for use by PCs in a LAN environment. It is an extension of the original Network Basic Input/Output System (NetBIOS) from IBM. NetBEUI uses a broadcast-based name to 802.x address translation mechanism. Because NetBEUI has no network layer, it is a nonroutable protocol.

The NetBIOS Frames Control Protocol (NBFCP) enables packets from a NetBEUI application to be transferred via a PPP connection. NetBEUI/PPP is supported in the access server and Cisco enterprise images only.

Using the Cisco IOS implementation, remote NetBEUI users can have access to LAN-based NetBEUI services. The PPP link becomes the ramp for the remote node to access NetBIOS services on the LAN. (See Figure 6.) An Logical Link Control, type 2 (LLC2) connection is set up between the remote access client and router, and a second LLC2 connection is set up between the router and the remote access (NetBEUI) server.

Figure 6 NetBEUI Connection



By supporting NetBEUI remote clients over PPP, Cisco routers function as a native NetBEUI dial-in router for remote NetBEUI clients. Thus, you can offer remote access to a NetBEUI network through asynchronous or ISDN connections.

To enable a remote access client using a NetBEUI application to connect with the remote router providing NetBEUI services, configure interfaces on the remote access client side and the remote router side by using the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# netbios nbf`	Enables NBFCP on each side of a NetBEUI connection.

To view NetBEUI connection information, use the following command in EXEC mode:

Command	Purpose
`Router> show nbf sessions`	Views NetBEUI connection information.

Performance Parameters

TCP Packet Headers Compression

You can compress the headers of your TCP/IP packets to reduce their size and thereby increase performance. Header compression is particularly useful on networks with a large percentage of small packets, such as those supporting many Telnet connections. This feature compresses only the TCP header, so it has no effect on UDP packets or other protocol headers. The TCP header compression technique, described fully in RFC 1144, is supported on serial lines using High-Level Data Link Control (HDLC) or PPP encapsulation. You must enable compression on both ends of a serial connection.

You can optionally specify outgoing packets to be compressed only when TCP incoming packets on the same interface are compressed. If you do not specify this option, the Cisco IOS software will compress all traffic. The default is no compression.

You can also specify the total number of header compression connections that can exist on an interface. You should configure one connection for each TCP connection through the specified interface.

Note When compression is enabled, fast switching is disabled. Fast processors can handle several fast interfaces, such as T1 lines, that are running header compression. However, you should think carefully about traffic characteristics in your network before compressing TCP headers. You might want to use the monitoring commands to help compare network utilization before and after enabling header compression.

TCP Connection Attempt Time

You can set the amount of time that the Cisco IOS software will wait to attempt to establish a TCP connection. In previous versions of the Cisco IOS software, the system would wait a fixed 30 seconds when attempting to make the connection. This amount of time is not enough in networks that have dialup asynchronous connections, such as a network consisting of dial-on-demand links that are implemented over modems, because it will affect your ability to use Telnet over the link (from the router) if the link must be brought up.

Because the connection attempt time is a host parameter, it does not pertain to traffic going through the router, just to traffic originated at it.

To set the TCP connection attempt time, use the following command in global configuration mode:

IPX Packet Headers Compression over PPP

The Cisco IOS software permits compression of IPX packet headers over various WAN media. There are two protocols for IPX compression on point-to-point links:

- CIPX, also known as Telebit style compression
- Shiva compression, which is proprietary

Cisco routers support IPX Header Compression (CIPX) on all point-to-point Novell interfaces over various WAN media.

CIPX is described in RFC 1553, *Compressing IPX Headers Over WAN Media*. The CIPX algorithm is based on the same concepts as Van Jacobson TCP/IP header compression algorithm. CIPX operates over PPP WAN links using either the IPXCP or IPXWAN communications protocols.

CIPX compresses all IPX headers and IPX/NCP headers for Novell packets with the following Network Control Program (NCP) packet types:

- 0x2222—NCP request from workstation
- 0x3333—NCP replies from file server

In this version of software, CIPX is configurable only for PPP links.

CIPX header compression can reduce header information from 30 bytes down to as little as 1 byte. This reduction can save bandwidth and reduce costs associated with IPX routing over WAN links that are configured to use IPXCP or IPXWAN.

Consider the following issues before implementing CIPX:

- CIPX is supported on all point-to-point IPX interfaces using PPP or IPXWAN processing (or both).
- CIPX needs to be negotiated for both directions of the link, because it uses the reverse direction of the link for communicating decompression problems back to the originating peer. In other words, all peer routers must have CIPX enabled.

Note We recommend that you keep a slot value of 16. Because slots are maintained in the router buffer, a larger number can impact buffer space for other operations.

Fast Switching

Fast switching involves the use of a high-speed switching cache for IP routing. With fast switching, destination IP addresses are stored in the high-speed cache so that some time-consuming table lookups can be avoided. The Cisco IOS software generally offers better packet transfer performance when fast switching is enabled.

Route Cache Invalidation

The high-speed route cache used by IP fast switching is invalidated when the IP routing table changes. By default, the invalidation of the cache is delayed slightly to avoid excessive CPU load while the routing table is changing.

Note This task normally should not be necessary. It should be performed only under the guidance of technical staff. Incorrect configuration can seriously degrade the performance of your router.

SLIP and PPP Banner Messages Customization

This feature enables you to customize the banner that is displayed when making a SLIP or PPP connection to avoid connectivity problems the default banner message causes in some non-Cisco SLIP and PPP dialup software. This feature is particularly useful when legacy client applications require a specialized connection string.

You can also use tokens in the banner message to display current IOS configuration variables. Tokens are keywords of the form **$(***token***)**. When you include tokens in a banner command, Cisco IOS will replace **$(***token***)** with the corresponding configuration variable.

Table 1 lists the tokens that you can use in the **banner slip-ppp** command.

Table 1 SLIP Banner Tokens

Tokens	Information Displayed in Banner
Global	
$(hostname)	Hostname of the router
$(domain)	Domain name of the router
Slip/PPP Banner-Specific	
$(peer-ip)	IP address of the peer machine
$(gate-ip)	IP address of the gateway machine
$(encap)	Encapsulation type (SLIP, PPP, and so on)
$(encap-alt)	Encapsulation type displayed as SL/IP instead of SLIP
$(mtu)	MTU size

How to Configure Asynchronous SLIP and PPP

To configure SLIP and PPP, perform the tasks in the following sections; all tasks are optional:

- Configuring Network-Layer Protocols over SLIP and PPP
- Asynchronous Host Mobility
- Additional Remote Node Connections
- Remote Access to NetBEUI Services
- Performance Parameters

Configuring Network-Layer Protocols over SLIP and PPP

Configuring IP and PPP

To enable IP-PPP (IPCP) on a synchronous or an asynchronous interface, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# ip address` *ip-address mask* `[secondary]`	Configures IP routing on the interface.
	or	
	`Router(config-if)# ip unnumbered` *type number*	Configures IP unnumbered routing on a serial interface.
Step 2	`Router(config-if)# encapsulation ppp`	Enables PPP encapsulation on the serial interface.
Step 3	`Router(config-if)# async mode interactive`	Enables interactive mode on an asynchronous interface.

Configuring IPX and PPP and Associating Asynchronous Interfaces with Loopback Interfaces

To permit IPX client connections to an asynchronous interface, the interface must be associated with a loopback interface configured to run IPX. To permit such connections, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing.
Step 2	`Router(config)# interface loopback number`	Creates a loopback interface, which is a virtual interface existing only inside the router, and begins interface configuration mode.
Step 3	`Router(config-if)# ipx network network`[1]	Enables IPX routing on the loopback interface.
Step 4	`Router(config-if)# exit`	Exits to global configuration mode.
Step 5	`Router(config)# interface async number`	Enters interface configuration mode for the asynchronous interface.
Step 6	`Router(config-if)# ip unnumbered type number`	Configures IP unnumbered routing on the interface.
Step 7	`Router(config-if)# encapsulation ppp`	Enables PPP encapsulation on the interface.
Step 8	`Router(config-if)# async mode interactive`	Enables interactive mode on an asynchronous interface.
Step 9	`Router(config-if)# ipx ppp-client loopback number`	Assigns the asynchronous interface to the loopback interface configured for IPX.
Step 10	`Router(config-if)# ipx update interval`	Turns off Service Advertising Protocol (SAP) updates to optimize bandwidth on asynchronous interfaces.

1. Every interface must have a unique IPX network number.

Configuring IPX and PPP Using Dedicated IPX Network Numbers for Each Interface

To enable IPX and PPP, use the following commands beginning in global configuration mode. The first five steps are required. The last step is optional.

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing.
Step 2	`Router(config)# interface loopback number`	Creates a loopback interface, which is a virtual interface existing only inside the router, and begins interface configuration mode.
Step 3	`Router(config-if)# encapsulation ppp`	Enables PPP encapsulation on the interface.
Step 4	`Router(config-if)# async mode interactive`	Enables interactive mode on an asynchronous interface.
Step 5	`Router(config-if)# ipx network network`[1]	Enables IPX routing on the interface.
Step 6	`Router(config-if)# ipx update interval`	(Optional) Turns off SAP updates to optimize bandwidth on asynchronous interfaces.

1. Every interface must have a unique IPX network number.

Enabling IPX and PPP over X.25 to an IPX Network on Virtual Terminal Lines

To enable IPX to run over your PPP sessions on virtual terminal lines, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing.
Step 2	`Router(config)# interface loopback number`	Creates a loopback interface and begins interface configuration mode.
Step 3	`Router(config-if)# ipx network network`[1]	Enables a virtual IPX network on the loopback interface.
Step 4	`Router(config-if)# vty-async ipx ppp-client loopback number`	Enables IPX-PPP on virtual terminal lines by assigning it to the loopback interface configured for IPX.

1. Every loopback interface must have a unique IPX network number.

Configuring AppleTalk and PPP

To enable ATCP for PPP, use the following commands in interface configuration (asynchronous) mode:

	Command	Purpose
Step 1	`Router(config-if)# encapsulation ppp`	Defines encapsulation as PPP on this interface.
Step 2	`Router(config-if)# appletalk virtual-net network-number zone-name`	Creates an internal network on the server.
Step 3	`Router(config-if)# appletalk client-mode`	Enables client-mode on this interface.

Configuring IP and SLIP

To enable IP-SLIP on a synchronous or asynchronous interface, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# ip address ip-address mask`	Configures IP routing on the interface.
	or	
	`Router(config-if)# ip unnumbered type number`	Configures IP unnumbered routing on a serial interface.
Step 2	`Router(config-if)# encapsulation slip`	Enables SLIP encapsulation on the serial interface.
Step 3	`Router(config-if)# async mode interactive`	Enables interactive mode on an asynchronous interface.

Configuring Asynchronous Host Mobility

To enable asynchronous host mobility, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip tcp async-mobility server`	Enables asynchronous listening on TCP tunnel port 57.

	Command	Purpose
Step 2	`Router(config)# exit`	Returns to user EXEC mode.
Step 3	`Router# tunnel host`	Sets up a network-layer connection to a router by specifying its Internet name or address. Replace the *host* argument with the name or address of the device that you want to connect to.

Configuring Performance Parameters

- TCP Packet Headers Compression
- TCP Connection Attempt Time
- IPX Packet Headers Compression over PPP
- Fast Switching
- Route Cache Invalidation
- SLIP and PPP Banner Messages Customization

Compressing TCP Packet Headers

To enable compression, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# ip tcp header-compression [passive]`	Enables TCP header compression.
Step 2	`Router(config-if)# ip tcp compression-connections number`	Specifies the total number of header compression connections that can exist on an interface.

Setting the TCP Connection Attempt Time

Command	Purpose
`Router(config)# ip tcp synwait-time seconds`	Sets the amount of time for which the Cisco IOS software will wait to attempt to establish a TCP connection.

Compressing IPX Packet Headers over PPP

To configure CIPX, use the following command in global configuration mode:

Command	Purpose
`Router(config)# ipx compression cipx number-of-slots`	Compresses IPX packet headers in a PPP session.

Enabling Fast Switching

To enable or disable fast switching, use the following commands in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# ip route-cache`	Enables fast-switching (use of a high-speed route cache for IP routing).
Step 2	`Router(config-if)# no ip route-cache`	Disables fast switching and enables load balancing on a per-packet basis.

Controlling Route Cache Invalidation

To control route cache invalidation, use the following commands in global configuration mode as needed for your network:

Note This task normally should not be necessary. It should be performed only under the guidance of technical staff. Incorrect configuration can seriously degrade the performance of your router.

	Command	Purpose
Step 1	`Router(config)# no ip cache-invalidate-delay`	Allows immediate invalidation of the cache.
Step 2	`Router(config)# ip cache-invalidate-delay [minimum maximum quiet-threshold]`	Delays invalidation of the cache.

Customizing SLIP and PPP Banner Messages

To configure the SLIP-PPP banner message, use the following command in global configuration mode:

Command	Purpose
`Router(config)# banner slip-ppp d message d`	Configures the SLIP-PPP banner to display a customized message.

Configuration Examples for Asynchronous SLIP and PPP

- Examples: Basic PPP Configurations
- Examples: Remote Node NetBEUI
- Example: Remote Network Access Using PPP Basic Configuration
- Example: Remote Network Access Using PPP and Routing IP
- Example: Remote Network Access Using a Leased Line with Dial-Backup and PPP
- Example: Multilink PPP Using Multiple Asynchronous Interfaces

Examples: Basic PPP Configurations

The following example illustrates how to make a connection when the system administrator defines a default IP address by including the **peer default ip address** command in interface configuration mode.

Note The **peer default ip address** command replaces the **async default ip address** command.

Once a correct password is entered, you are placed in SLIP mode, and the IP address appears:

```
Router> slip
Password:
Entering SLIP mode.
Your IP address is 192.168.7.28, MTU is 1524 bytes
```

The following example shows the prompts displayed and the response required when dynamic addressing is used to assign the SLIP address:

```
Router> slip
IP address or hostname? 192.168.6.15
Password:
Entering SLIP mode
Your IP address is 192.168.6.15, MTU is 1524 bytes
```

In the previous example, the address 192.168.6.15 had been assigned as the default. Password verification is still required before SLIP mode can be enabled, as follows:

```
Router> slip default
Password:
Entering SLIP mode
Your IP address is 192.168.6.15, MTU is 1524 bytes
```

The following example illustrates the implementation of header compression on the interface with the IP address 172.16.2.1:

```
Router> slip 172.16.2.1 /compressed
Password:
Entering SLIP mode.
Interface IP address is 172.16.2.1, MTU is 1500 bytes.
Header compression will match your system.
```

In the preceding example, the interface is configured for **ip tcp header-compression passive**, which permitted the user to enter the **/compressed** keyword at the EXEC mode prompt. The message "Header compression will match your system" indicates that the user has specified compression. If the line was configured for **ip tcp header-compression on**, this line would read "Header compression is On."

The following example specifies a TACACS server named parlance for address authentication:

```
Router> slip 10.0.0.1@parlance
Password:
Entering SLIP mode.
Interface IP address is 10.0.0.1, MTU is 1500 bytes
Header compression will match your system.
```

The following example sets the SLIP-PPP banner using several tokens and the percent sign (%) as the delimiting character:

```
Router(config)# banner slip-ppp %
Enter TEXT message.  End with the character '%'.
Starting $(encap) connection from $(gate-ip) to $(peer-ip) using a maximum packet size of
$(mtu) bytes... %
```

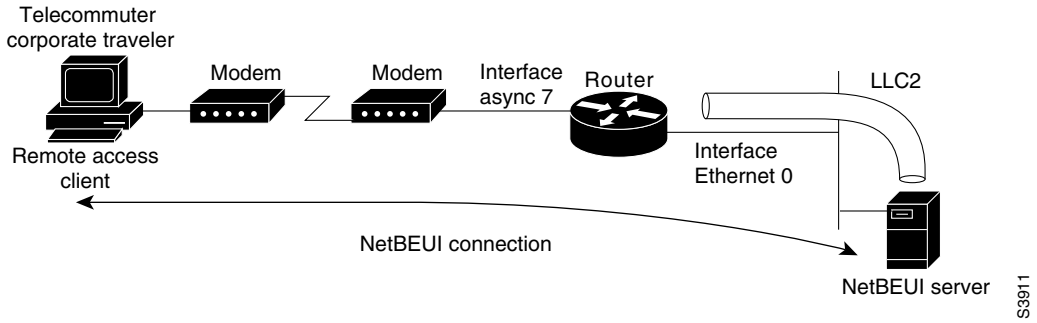
When you enter the **slip** command, you will see the following banner. Notice that the **$(***token***)** syntax is replaced by the corresponding configuration variables.

```
Starting SLIP connection from 192.168.69.96 to 172.16.80.8 using a maximum packet size of
1500 bytes...
```

Examples: Remote Node NetBEUI

In the following example, asynchronous interface 7 and Ethernet interface 0 are configured to enable NetBEUI connectivity between the corporate telecommuter client and the remote access (NetBEUI) server. The PC client is running the Chat legacy application in Windows NT to connect with the remote server. (See Figure 7.)

Figure 7 Connecting a Remote NetBEUI Client to a Server Through a Router



The configuration for the router is as follows:

```
interface async 7
 netbios nbf
 encapsulation ppp
```

You would also need to configure security, such as TACACS+, RADIUS, or another form of login authentication on the router.

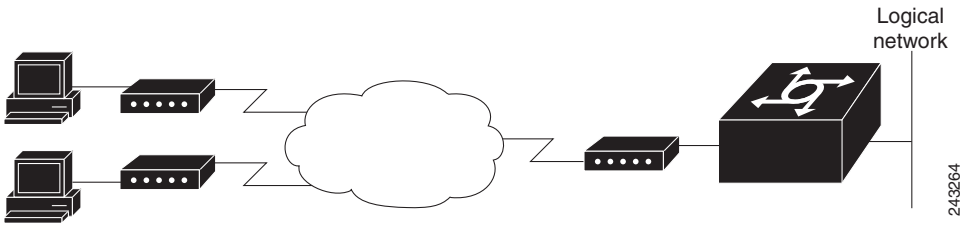
Example: Remote Network Access Using PPP Basic Configuration

Figure 8 illustrates a simple network configuration that includes remote PCs with modems connected via modem to a router. The cloud is a Public Switched Telephone Network (PSTN). The modems are connected via asynchronous lines, and the access server is connected to a local network.

In this example, the following is configured:

- An asynchronous line on the access server configured to use PPP encapsulation.
- An interface on the access server for the modem connection; this interface also needs to be configured to accept incoming modem calls.
- A default IP address for each incoming line.

Figure 8 Remote Network Access Using PPP



This default address indicates the address of the remote PC to the server, unless the user explicitly specifies another when starting the PPP session.

The server is configured for interactive mode with autoselect enabled, which allows the user to automatically begin a PPP session upon detection of a PPP packet from the remote PC; or, the remote PC can explicitly begin a PPP session by entering the **ppp** EXEC command at the prompt.

The configuration is as follows:

```
ip routing
!
interface ethernet 0
 ip address 192.168.32.12 255.255.255.0
!
interface async 1
 encapsulation ppp
 async mode interactive
 async default ip address 192.168.32.51
 async dynamic address
 ip unnumbered ethernet 0

line 1
 autoselect ppp
 modem callin
 speed 19200
```

Example: Remote Network Access Using PPP and Routing IP

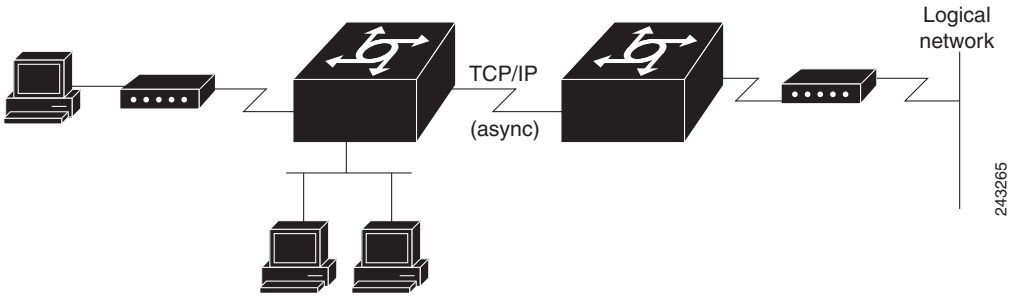
Figure 9 illustrates a network configuration that provides routing functionality, allowing routing updates to be passed across the asynchronous lines.

This network is composed of remote and local PCs connected via modem and network connections to an access server. This access server is connected to a second access server via an asynchronous line running TCP/IP. The second access server is connected to a local network via modem.

For this scenario, you will need to configure the following:

- An asynchronous line on both access servers configured to use PPP encapsulation
- An interface on both access servers for the modem connection and for this interface to be configured to accept incoming modem calls
- A default IP address for each incoming line
- IP routing on all configured interfaces

Figure 9 Routing on an Asynchronous Line Using PPP



The configuration is as follows:

```
interface async 1
 encapsulation ppp
 async mode interactive
 async default ip address 192.168.32.10
 async dynamic address
 ip unnumbered ethernet 0
 async dynamic routing
```

If you want to pass IP routing updates across the asynchronous link, enter the following commands:

```
line 1
 autoselect ppp
 modem callin
 speed 19200
```

Next, enter the following commands to configure the asynchronous lines between the access servers beginning in global configuration mode:

```
interface async 2
 async default ip address 192.168.32.55
 ip tcp header compression passive
```

Finally, configure routing as described in the *Cisco IOS IP Configuration Guide* using one of the following methods. The server can route packets three different ways.

- Use ARP, which is the default behavior.
- Use a default-gateway by entering the command **ip default-gateway** *x.x.x.x,* where *x.x.x.x* is the IP address of a locally attached router.
- Run an IP routing protocol such as Routing Information Protocol (RIP), Interior Gateway Routing Protocol (IGRP), Enhanced IGRP (EIGRP), or Open Shortest Path First (OSPF).

Example: Remote Network Access Using a Leased Line with Dial-Backup and PPP

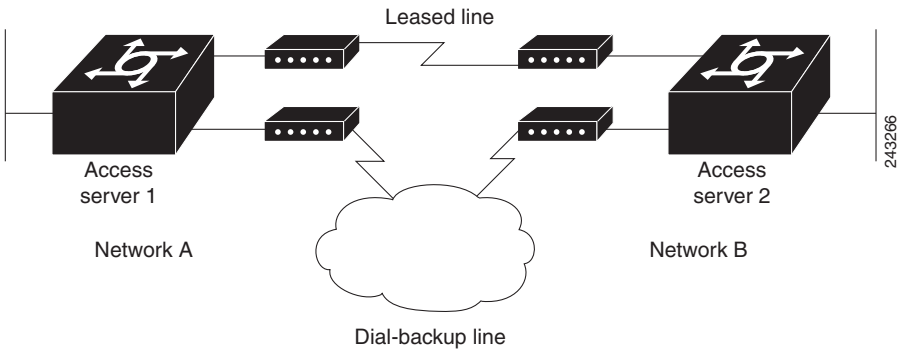
Figure 10 illustrates a scenario where two networks are connected via access servers on a leased line. Redundancy is provided by a dial-backup line over the PSTN so that if the primary leased line goes down, the dial-backup line will be automatically brought up to restore the connection. This configuration would be useful for using an auxiliary port as the backup port for a synchronous port.

For this scenario, you would need to configure the following:

- Two asynchronous interfaces on each access server

- Two modem interfaces
- A default IP address for each interface
- Dial-backup on one modem interface per access server
- An interface connecting to the related network of an access server

Figure 10 Asynchronous Leased Line with Backup



The configuration for this scenario follows:

```
hostname routerA
!
username routerB password cisco
chat-script backup "" "AT" TIMEOUT 30 OK atdt\T TIMEOUT 30 CONNECT \c !
!
interface Serial0
 backup interface Async1
 ip address 192.168.222.12 255.255.255.0
!
interface Async1
 ip address 172.16.199.1 255.255.255.0
 encapsulation ppp
 async default ip address 172.16.199.2
 async dynamic address
 async dynamic routing
 async mode dedicated
 dialer in-band
 dialer map IP 172.16.199.2 name routerB modem-script backup broadcast 3241129
 dialer-group 1
 ppp authentication chap
!
 dialer-list 1 protocol ip permit
!
line aux 0
 modem InOut
 rxspeed 38400
 txspeed 38400
```

Example: Multilink PPP Using Multiple Asynchronous Interfaces

The following example shows how to configure MLP using multiple asynchronous interfaces:

```
chat-script backup "" "AT" TIMEOUT 30 OK atdt\T TIMEOUT 30 CONNECT \c
!
ip address-pool local
```

```
ip pool foo 10.0.1.5 10.0.1.15
!
int as 1 (2, 3)
 no ip address
 dialer in-band
 encapsulation ppp
 ppp multilink
 dialer-rotary 1
!
interface dialer 1
 encaps ppp
 ip unnumbered ethernet 0
 peer default ip addr pool foo
 ppp authentication chap
 ppp multilink
 dialer in-band
 dialer map ip 10.200.100.9 name WAN-R3 modem-script backup broadcast 2322036
 dialer load-threshold 5 either
 dialer-group 1
!
dialer-list 1 protocol ip permit
!
line line 1 3
modem InOut
speed 115000
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Master Commands List, All Releases*
Dial Technologies commands	*Dial Technologies Command Reference*

Standards

Standard	Title
RFC 1055	
RFCs 1331	
RFCs 1332	
RFC 1084	

MIBs

MIB	MIBs Link
•	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring Asynchronous SLIP and PPP

<<Table 2>> lists the release history for this feature.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 2 lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 2 Feature Information for Configuring Asynchronous SLIP and PPP

Feature Name	Releases	Feature Information
PPP	11.2 12.2(28)SB 12.2(33)SRC 12.2(33)XNA 15.2(2)S	The Configuring Asynchronous SLIP and PPP module describes how to configure asynchronous Serial Line Internet Protocol (SLIP) and PPP. The following commands were introduced or modified: **ppp multilink, ppp multilink group.**

Configuring Media-Independent PPP and Multilink PPP

First Published: May 10, 2001
Last Updated: November 20, 2014

The Configuring Media-Independent PPP and Multilink PPP module describes how to configure PPP and Multilink PPP (MLP) features on any interface. This module also describes address pooling for point-to-point links, which is available on all asynchronous serial, synchronous serial, and ISDN interfaces.

Multilink PPP provides a method for spreading traffic across multiple physical WAN links.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Configuring Media-Independent PPP and Multilink PPP" section on page 50.

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

Contents

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

Prerequisites for Media-Independent PPP and Multilink PPP

Understanding PPP and multilink operations.

Information About Media-Independent PPP and Multilink PPP

To configure the Media-Independent PPP and Multilink PPP, you should understand the following concepts:

Point-to-Point Protocol

Point-to-Point Protocol (PPP), described in RFC 1661, encapsulates network layer protocol information over point-to-point links. You can configure PPP on the following types of physical interfaces:

- Asynchronous serial
- High-Speed Serial Interface (HSSI)
- ISDN
- Synchronous serial

The implementation of PPP supports authentication using Challenge Handshake Authentication Protocol (CHAP), Microsoft Challenge Handshake Authentication Protocol (MS-CHAP), or Password Authentication Protocol (PAP), and the option 4, and option 5, and Magic Number configuration options.

Magic Number support is available on all serial interfaces. PPP always attempts to negotiate for Magic Numbers, which are used to detect looped-back lines. Depending on how the **down-when-looped** command is configured, the router might shut down a link if it detects a loop.

CHAP or PPP Authentication

PPP with CHAP or PAP authentication is often used to inform the central site about which remote routers are connected to it.

With this authentication information, if the router or access server receives another packet for a destination to which it is already connected, it does not place an additional call. However, if the router or access server is using rotaries, it sends the packet out the correct port.

CHAP and PAP were originally specified in RFC 1334, and CHAP was updated in RFC 1994. These protocols are supported on synchronous and asynchronous serial interfaces. When using CHAP or PAP authentication, each router or access server identifies itself by a *name*. This identification process prevents a router from placing another call to a router to which it is already connected, and also prevents unauthorized access.

Access control using CHAP or PAP is available on all serial interfaces that use PPP encapsulation. The authentication feature reduces the risk of security violations on your router or access server. You can configure either CHAP or PAP for the interface.

Note To use CHAP or PAP, you must be running PPP encapsulation.

When CHAP is enabled on an interface and a remote device attempts to connect to it, the local router or access server sends a CHAP packet to the remote device. The CHAP packet requests or "challenges" the remote device to respond. The challenge packet consists of an ID, a random number, and the hostname of the local router.

The required response has two parts:

- An encrypted version of the ID, a secret password, and the random number
- Either the hostname of the remote device or the name of the user on the remote device

When the local router or access server receives the response, it verifies the secret password by performing the same encryption operation as indicated in the response and looking up the required hostname or username. The secret passwords must be identical on the remote device and the local router.

Because this response is sent, the password is never sent in clear text, preventing other devices from stealing it and gaining illegal access to the system. Without the proper response, the remote device cannot connect to the local router.

CHAP transactions occur only when a link is established. The local router or access server does not request a password during the rest of the call. (The local device can, however, respond to such requests from other devices during a call.)

When PAP is enabled, the remote router attempting to connect to the local router or access server is required to send an authentication request. The username and password specified in the authentication request are accepted, and the Cisco IOS software sends an authentication acknowledgment.

After you have enabled CHAP or PAP, the local router or access server requires authentication from remote devices. If the remote device does not support the enabled protocol, no traffic will be passed to that device.

To use CHAP or PAP:

- Enable PPP encapsulation.
- Enable CHAP or PAP on the interface.

For CHAP, configure hostname authentication and the secret password for each remote system with which authentication is required.

Microsoft Point-to-Point Compression

Microsoft Point-to-Point Compression (MPPC) is a scheme used to compress PPP packets between Cisco and Microsoft client devices. The MPPC algorithm is designed to optimize bandwidth utilization in order to support multiple simultaneous connections. The MPPC algorithm uses a Lempel-Ziv (LZ)-based algorithm with a continuous history buffer called a dictionary.

The Compression Control Protocol (CCP) configuration option for MPPC is 18.

Exactly one MPPC datagram is encapsulated in the PPP information field. The PPP protocol field indicates the hexadecimal type of 00FD for all compressed datagrams. The maximum length of the MPPC datagram sent over PPP is the same as the MTU of the PPP interface; however, this length cannot be greater than 8192 bytes because the history buffer is limited to 8192 bytes. If compressing the data results in data expansion, the original data is sent as an uncompressed MPPC packet.

The history buffers between compressor and decompressor are synchronized by maintaining a 12-bit coherency count. If the decompressor detects that the coherency count is out of sequence, the following error recovery process is performed:

1. The Reset Request (RR) packet is sent from the decompressor.
2. The compressor then flushes the history buffer and sets the flushed bit in the next packet it sends.
3. Upon receiving the flushed bit set packet, the decompressor flushes the history buffer.

Synchronization is achieved without CCP using the Reset Acknowledge (RA) packet, which can consume additional time.

Compression negotiation between a router and a Windows 95 client occurs through the following process:

1. Windows 95 sends a request for both STAC (option 17) and MPPC (option 18) compression.
2. The router sends a negative acknowledgment (NAK) requesting only MPPC.
3. Windows 95 resends the request for MPPC.

The router sends an acknowledgment (ACK) confirming MPPC compression negotiation.

IP Address Pooling

A point-to-point interface must be able to provide a remote node with its IP address through the IP Control Protocol (IPCP) address negotiation process. The IP address can be obtained from a variety of sources. The address can be configured through the command line, entered with an EXEC-level command, provided by TACACS+ or the Dynamic Host Configuration Protocol (DHCP), or from a locally administered pool.

IP address pooling uses a pool of IP addresses from which an incoming interface can provide an IP address to a remote node through IPCP address negotiation process. IP address pooling also enhances configuration flexibility by allowing multiple types of pooling to be active simultaneously.

The IP address pooling feature allows configuration of a global default address pooling mechanism, per-interface configuration of the address pooling mechanism, and per-interface configuration of a specific address or pool name.

Peer Address Allocation

A peer IP address can be allocated to an interface through several methods:

- Dialer map lookup—This method is used only if the peer requests an IP address, no other peer IP address has been assigned, and the interface is a member of a dialer group.
- PPP or Serial Line Internet Protocol (SLIP) EXEC command—An asynchronous dialup user can enter a peer IP address or hostname when PPP or SLIP is invoked from the command line. The address is used for the current session and then discarded.
- IPCP negotiation—If the peer presents a peer IP address during IPCP address negotiation and no other peer address is assigned, the presented address is acknowledged and used in the current session.
- Default IP address.
- TACACS+ assigned IP address—During the authorization phase of IPCP address negotiation, TACACS+ can return an IP address that the user being authenticated on a dialup interface can use. This address overrides any default IP address and prevents pooling from taking place.

- DHCP retrieved IP address—If configured, the routers acts as a proxy client for the dialup user and retrieves an IP address from a DHCP server. That address is returned to the DHCP server when the timer expires or when the interface goes down.
- Local address pool—The local address pool contains a set of contiguous IP addresses (a maximum of 1024 addresses) stored in two queues. The free queue contains addresses available to be assigned and the used queue contains addresses that are in use. Addresses are stored to the free queue in first-in, first-out (FIFO) order to minimize the chance the address will be reused, and to allow a peer to reconnect using the same address that it used in the last connection. If the address is available, it is assigned; if not, another address from the free queue is assigned.
- Chat script (asynchronous serial interfaces only)—The IP address in the **dialer map** command entry that started the script is assigned to the interface and overrides any previously assigned peer IP address.
- Virtual terminal/protocol translation—The translate command can define the peer IP address for a virtual terminal (pseudo asynchronous interface).
- The pool configured for the interface is used, unless TACACS+ returns a pool name as part of authentication, authorization, and accounting (AAA). If no pool is associated with a given interface, the global pool named default is used.

Precedence Rules

The following precedence rules of peer IP address support determine which address is used. Precedence is listed from most likely to least likely:

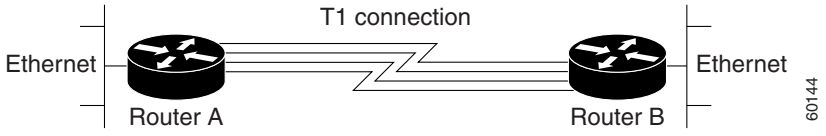
1. AAA/TACACS+ provided address or addresses from the pool named by AAA/TACACS+
2. An address from a local IP address pool or DHCP (typically not allocated unless no other address exists)
3. Dialer map lookup address (not done unless no other address exists)
4. Address from an EXEC-level PPP or SLIP command, or from a chat script
5. Configured address from the **peer default ip address** command or address from the protocol **translate** command
6. Peer-provided address from IPCP negotiation (not accepted unless no other address exists)

MLP on Synchronous Serial Interfaces

Address pooling is available on all asynchronous serial, synchronous serial, ISDN BRI, and ISDN PRI interfaces that are running PPP and PPPoX sessions.

MLP provides characteristics are most similar to hardware inverse multiplexers, with good manageability and Layer 3 services support. Figure 1 shows a typical inverse multiplexing application using two Cisco routers and Multilink PPP over four T1 lines.

Figure 1 Inverse Multiplexing Application Using Multilink PPP



How to Configure Media-Independent PPP and Multilink PPP

This section includes the following procedures:

Configuring PPP and MLP

Perform the following task in interface configuration mode to configure PPP on a serial interface (including ISDN). This task is required for PPP encapsulation.

You can also complete the tasks in the following sections; these tasks are optional but offer a variety of uses and enhancements for PPP on your systems and networks:

See the "Monitoring and Maintaining PPP and MLP Interfaces" section on page 35 for tips on maintaining PPP. See the "Configuration Examples for PPP and MLP" section on page 36 to understand how to implement PPP and MLP in your network.

Enabling PPP Encapsulation

The **encapsulation ppp** command enables PPP on serial lines to encapsulate IP and other network protocol datagrams.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **configure fastethernet** *number*
4. **encapsulation ppp**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface fastethernet** *number* **Example:** `Device(config)# interface fastethernet 0/0`	Enters interface configuration mode.
Step 4	**encapsulation ppp** **Example:** `Device(config-if) # encapsulation ppp`	Enables PPP encapsulation. **Note** PPP echo requests are used as keepalives to minimize disruptions to the end users of your network. Use the **no keepalive command** to disable echo requests.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode.

Enabling CHAP or PAP Authentication

To enable CHAP or PAP authentication, perform the steps mentioned in this section.

Caution If you use a list name that has not been configured with the **aaa authentication ppp** command, you disable PPP on the line.

For an example of CHAP, see the section ""Examples: CHAP with an Encrypted Password:" section on page 36". CHAP is specified in RFC 1994, *PPP Challenge Handshake Authentication Protocol (CHAP)*.

For information about MS-CHAP, see *MS-CHAP Support*.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface fastethernet** *number*
4. **ppp authentication** {**chap** | **chap pap** | **pap chap** | **pap**} [**if-needed**] [*list-name* | **default**] [**callin**]
5. **ppp use-tacacs** [**single-line**]

   or

   aaa authentication ppp

6. **exit**
7. **username** *name* [**user-maxlinks** *link-number*] **password** *secret*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface fastethernet** *number* **Example:** `Device(config)# interface fastethernet 0/0`	Enters Interface Configuration mode.
Step 4	**ppp authentication** {**chap** | **chap pap** | **pap chap** | **pap**} [**if-needed**] [*list-name* | **default**] [**callin**] **Example:** `Device(config-if)# ppp authentication chap`	Defines the authentication methods supported and the order in which they are used. **Note** Use the **ppp authentication chap** command only with TACACS or extended TACACS. **Note** With AAA configured on the router and list names defined for AAA, the *list-name* optional argument can be used with AAA/TACACS+. Use the **ppp use-tacacs** command with TACACS and Extended TACACS. Use the **aaa authentication ppp** command with AAA/TACACS+.

Step 5	**ppp use-tacacs** [**single-line**]
	or
	aaa authentication ppp
	
	Example:
	`Device(config-if)# ppp use-tacacs single-line`
	or
	`Device(config-if)# aaa authentication ppp`
		Configure TACACS on a specific interface as an alternative to global host authentication.

Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 7	**username** *name* [**user-maxlinks** *link-number*] **password** *secret* **Example:** `Device(config)# username name user-maxlinks 1 password password1`	Configures identification. • Optionally, you can specify the maximum number of connections a user can establish. • To use the **user-maxlinks** keyword, you must also use the **aaa authorization network default local** command and PPP encapsulation and name authentication on all the interfaces the user will be accessing.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Compression of PPP Data

You can configure point-to-point software compression on serial interfaces that use PPP encapsulation. Compression reduces the size of a PPP frame via lossless data compression. PPP encapsulations support both predictor and Stacker compression algorithms.

If most of your traffic is already compressed files, do not use compression.

To configure compression of PPP data, perform the steps in this section.

Software Compression

Software compression is available on all router platforms. Software compression is performed by the main processor in the router.

Compression is performed in software and might significantly affect system performance. We recommend that you disable compression if the router CPU load exceeds 65 percent. To display the CPU load, use the **show process cpu** EXEC command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface fastethernet** *number*
4. **encapsulation PPP**
5. **compress** [**predictor | stac | mppc** [**ignore-pfc**]]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface fastethernet** *number* **Example:** `Device(config)# interface fastethernet 0/0`	Enters interface configuration mode.
Step 4	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables encapsulation of a single protocol on the serial line.
Step 5	**compress** [**predictor** | **stac** | **mppc** [**ignore-pfc**]] **Example:** `Device(config-if)# compress predictor`	Enables compression.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode.

Configuring Microsoft Point-to-Point Compression

Perform this task to configure MPCC. This will help you set MPPC once PPP encapsulation is configured on the router.

Prerequisites

Ensure that PPP encapsulation is enabled before you configure MPPC. For information on how to configure PPP encapsulation, see the "Enabling PPP Encapsulation" section on page 6".

Restrictions

The following restrictions apply to the MPPC feature:

- MPPC is supported only with PPP encapsulation.
- Compression can be processor intensive because it requires a reserved block of memory to maintain the history buffer. Do not enable modem or hardware compression because it may cause performance degradation, compression failure, or data expansion.
- Both ends of the point-to-point link must be using the same compression method (STAC, Predictor, or MPPC, for example).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *number*
4. **compress** [**mppc** [**ignore-pfc**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *number* **Example:** `Device(config)# interface serial 2/0`	Enters interface configuration mode.
Step 4	**compress** [**mppc** [**ignore-pfc**]] **Example:** `Device(config-if)# compress mppc`	Enables encapsulation of a single protocol on the serial line. • The **ignore-pfc** keyword instructs the router to ignore the protocol field compression flag negotiated by Link Control Protocol (LCP). For example, the uncompressed standard protocol field value for IP is 0x0021 and 0x21 when compression is enabled. When the **ignore-pfc** option is enabled, the router will continue to use the uncompressed value (0x0021). Using the **ignore-pfc** option is helpful for some asynchronous driver devices that use an uncompressed protocol field (0x0021), even though the protocol field compression is negotiated between peers.

Examples

Following is sample **debug ppp negotiation** command output showing protocol reject:

```
PPP Async2: protocol reject received for protocol = 0x2145
PPP Async2: protocol reject received for protocol = 0x2145
PPP Async2: protocol reject received for protocol = 0x2145
```

Configuring IP Address Pooling

You can define the type of IP address pooling mechanism used on router interfaces in one or both of the ways described in the following sections:

Defining the Global Default Address Pooling Mechanism

The global default mechanism applies to all point-to-point interfaces that support PPP encapsulation and that have not otherwise been configured for IP address pooling. You can define the global default mechanism to be either DHCP or local address pooling.

To configure the global default mechanism for IP address pooling, perform the tasks in the following sections:

After you have defined a global default mechanism, you can disable it on a specific interface by configuring the interface for some other pooling mechanism. You can define a local pool other than the default pool for the interface or you can configure the interface with a specific IP address to be used for dial-in peers.

You can also control the DHCP network discovery mechanism; see the following section for more information:

Defining DHCP as the Global Default Mechanism

DHCP specifies the following components:

- A DHCP server—A host-based DHCP server configured to accept and process requests for temporary IP addresses.
- A DHCP proxy client—A Cisco access server configured to arbitrate DHCP calls between the DHCP server and the DHCP client. The DHCP client-proxy feature manages a pool of IP addresses available to dial-in clients without a known IP address.

Perform this task to enable DHCP as the global default mechanism.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip address-pool dhcp-proxy-client**
4. **ip dhcp-server** [*ip-address* | *name*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip address-pool dhcp-proxy-client** **Example:** `Device(config)# ip address-pool dhcp-proxy-client`	Specifies the DHCP client-proxy feature as the global default mechanism. • The **peer default ip address** command and the **member peer default ip address** command can be used to define default peer IP addresses. **Note** You can provide as few as one or as many as ten DHCP servers for the proxy client (the Cisco router or access server) to use. The DHCP servers provide temporary IP addresses.
Step 4	**ip dhcp-server** [*ip-address* | *name*] **Example:** `Device(config)# ip dhcp-server 209.165.201.1`	(Optional) Specifies the IP address of a DHCP server for the proxy client to use.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

Defining Local Address Pooling as the Global Default Mechanism

Perform this task to define local address pooling as the global default mechanism.

Note If no other pool is defined, a local pool called "default" is used. Optionally, you can associate an address pool with a named pool group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip address-pool local**
4. **ip local pool** {*named-address-pool* | **default**} *first-IP-address* [*last-IP-address*] [**group** *group-name*] [**cache-size** *size*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip address-pool local** **Example:** `Device(config)# ip address-pool local`	Specifies local address pooling as the global default mechanism.
Step 4	**ip local pool** {*named-address-pool* | d**efault**} *first-IP-address* [*last-IP-address*] [**group** *group-name*] [**cache-size** *size*] **Example:** `Device(config)# ip local pool default 192.0.2.1`	Creates one or more local IP address pools.

Controlling DHCP Network Discovery

Perform the steps in this section to allow peer routers to dynamically discover Domain Name System (DNS) and NetBIOS name server information configured on a DHCP server using PPP IPCP extensions.

The **ip dhcp-client network-discovery** global configuration command provides a way to control the DHCP network discovery mechanism. The number of DHCP Inform or Discovery messages can be set to 1 or 2, which determines how many times the system sends the DHCP Inform or Discover messages before stopping network discovery. You can set a timeout period from 3 to 15 seconds, or leave the default timeout period at 15 seconds. The default for the **informs** and **discovers** keywords is 0, which disables the transmission of these messages.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dhcp-client network-discovery informs** *number-of-messages* **discovers** *number-of-messages* **period** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp-client network-discovery informs** *number-of-messages* **discovers** *number-of-messages* **period** *seconds* **Example:** `Device(config)# ip dhcp-client network-discovery informs 2 discovers 2 period 2`	Provides control of the DHCP network discovery mechanism by allowing the number of DHCP Inform and Discover messages to be sent, and a timeout period for retransmission, to be configured.

Configuring IP Address Assignment

Perform this task to configure IP address allignment.

After you have defined a global default mechanism for assigning IP addresses to dial-in peers, you can configure the few interfaces for which it is important to have a nondefault configuration. You can do any of the following;

- Define a nondefault address pool for use by a specific interface.
- Define DHCP on an interface even if you have defined local pooling as the global default mechanism.
- Specify one IP address to be assigned to all dial-in peers on an interface.
- Make temporary IP addresses available on a per-interface basis to asynchronous clients using SLIP or PPP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip local pool** {**named-address-pool** | **default**} {*first-IP-address* [*last-IP-address]*} [**group** *group-name*] [**cache-size** *size*]}
4. **interface** *type number*
5. **peer default ip address pool** *pool-name-list*
6. **peer default ip address pool dhcp**
7. **peer default ip address** *ip-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip local pool** {*named-address-pool* | d**efault**} {*first-IP-address* [*last-IP-address*]} [**group** *group-name*] [**cache-size** *size*]} **Example:** `Device(config)# ip local pool default 192.0.2.0`	Creates one or more local IP address pools.
Step 4	**interface** *type number* **Example:** `Device(config)# interface ethernet 2/0`	Specifies the interface and enters interface configuration mode.
Step 5	**peer default ip address pool** *pool-name-list* **Example:** `Device(config-if)# peer default ip address pool 2`	Specifies the pool or pools for the interface to use.
Step 6	**peer default ip address pool dhcp** **Example:** `Device(config-if)# peer default ip address pool dhcp`	Specifies DHCP as the IP address mechanism on this interface.
Step 7	**peer default ip address** *ip-address* **Example:** `Device(config-if)# peer default ip address 192.0.2.2`	Specifies the IP address to assign to all dial-in peers on an interface.

Troubleshooting PPP

You can troubleshoot PPP reliable link by using the **debug lapb** command and the **debug ppp negotiations**, **debug ppp errors**, and **debug ppp packets** commands. You can determine whether Link Access Procedure, Balanced (LAPB) has been established on a connection by using the **show interface** command.

Disabling or Reenabling Peer Neighbor Routes

The Cisco IOS software automatically creates neighbor routes by default; that is, it automatically sets up a route to the peer address on a point-to-point interface when the PPP IPCP negotiation is completed.

To disable this default behavior or to reenable it once it has been disabled, perform the following task:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no peer neighbor-route**
5. **peer neighbor-route**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/1`	Specifies the interface and enters interface configuration mode.
Step 4	**no peer neighbor-route** **Example:** `Device(config-if)# no peer neighbor-route`	Disables creation of neighbor routes.
Step 5	**peer neighbor-route** **Example:** `Device(config-if)# peer neighbor-route`	Reenables creation of neighbor routes. **Note** If entered on a dialer or asynchronous group interface, this command affects all member interfaces.

Configuring Multilink PPP

The Multilink PPP feature provides load balancing functionality over multiple WAN links, while providing multivendor interoperability, packet fragmentation and proper sequencing, and load calculation on both inbound and outbound traffic. The Cisco implementation of MLP supports the fragmentation and packet sequencing specifications in RFC 1990. Additionally, you can change the default endpoint discriminator value that is supplied as part of user authentication. Refer to RFC 1990 for more information about the endpoint discriminator.

MLP allows packets to be fragmented and the fragments to be sent at the same time over multiple point-to-point links to the same remote address. The multiple links come up in response to a defined dialer load threshold. The load can be calculated on inbound traffic, outbound traffic, or on either, as needed for the traffic between the specific sites. MLP provides bandwidth on demand and reduces transmission latency across WAN links.

MLP is designed to work over synchronous and asynchronous serial and BRI and PRI types of single or multiple interfaces that have been configured to support both dial-on-demand rotary groups and PPP encapsulation.

Perform the tasks in the following sections, as required for your network, to configure MLP:

- Configuring MLP on Synchronous Interfaces, page 18
- Configuring MLP on Asynchronous Interfaces, page 19
- Configuring MLP on a Single ISDN BRI Interface, page 21
- Configuring MLP on Multiple ISDN BRI Interfaces, page 23
- Configuring MLP Using Multilink Group Interfaces, page 25
- Changing the Default Endpoint Discriminator, page 27

Configuring MLP on Synchronous Interfaces

To configure Multilink PPP on synchronous interfaces, you configure the synchronous interfaces to support PPP encapsulation and Multilink PPP.

Perform this task to configure a synchronous interface.

SUMMARY STEPS

1. **enable**
2. **configuration terminal**
3. **interface serial** 1
4. **no ip address**
5. **encapuslation ppp**
6. **ppp multilink**
7. **pulse-time** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface serial** *number* **Example:** `Device(config)# interface serial 1`	Specifies an asynchronous interface and enters interface configuration mode.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Specifies no IP address for the interface.
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 6	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.
Step 7	**pulse-time** *seconds* **Example:** `Device(config-if)# pulse-time 60`	Enables pulsing data terminal ready (DTR) signal intervals on an interface. **Note** Repeat these steps for additional synchronous interfaces, as needed.

Configuring MLP on Asynchronous Interfaces

Perform the following steps in this section to configure an asynchronous interface to support DDR and PPP encapsulation and then configure a dialer interface to support PPP encapsulation, bandwidth on demand, and Multilink PPP.

At some point, adding more asynchronous interfaces does not improve performance, With the default maximum transmission unit (MTU) size, MLP should support three asynchronous interfaces using V.34 modems. However, packets might be dropped occasionally if the maximum transmission unit (MTU) size is small or large bursts of short frames occur.

Note To configure a dialer interface to support PPP encapsulation and Multilink PPP, use the **dialer load-threshold** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface async** *number*
4. **no ip address**
5. **dialer in-band**
6. **dialer rotary-group** *number*
7. **dialer load-threshold** *load* [**inbound** | **outbound** | **either**]

8. **ppp multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface async** *number* **Example:** `Device(config)# interface async 0/0`	Specifies an asynchronous interface and enters interface configuration mode.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Specifies no IP address for the interface.
Step 5	`Device(config-if)#` **encapsulation ppp** **Example:** `Device# configure terminal`	Enables PPP encapsulation.
Step 6	**dialer in-band** **Example:** `Device(config-if)# encapsulation ppp`	Enables DDR on the interface.
Step 7	**dialer rotary-group** *number* **Example:** `Device(config-if)# dialer rotary-group 1`	Includes the interface in a specific dialer rotary group.
Step 8	**dialer load-threshold** *load* [**inbound** | **outbound** | **either**] **Example:** `Device(config-if)# dialer load-threshold 100`	Configures bandwidth on demand by specifying the maximum load before the dialer places another call to a destination.
Step 9	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.

Configuring MLP on a Single ISDN BRI Interface

To enable MLP on a single ISDN BRI interface, you are not required to define a dialer rotary group separately because ISDN interfaces are dialer rotary groups by default.

Perform this task to enable PPP on an ISDN BRI interface.

If you do not use PPP authentication procedures (Step 8), your telephone service must pass caller ID information.

The load threshold number is required. For an example of configuring MLP on a single ISDN BRI interface, see the "Example: MLP on One ISDN BRI Interface" section on page 43.

Note When MLP is configured and you want a multilink bundle to be connected indefinitely, use the **dialer idle-timeout** command to set a high idle timer. The **dialer-load threshold 1** command does not keep a multilink bundle of *n* links connected indefinitely, and the **dialer-load threshold 2** command does not keep a multilink bundle of two links connected indefinitely.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface bri** *number*
4. **ip address** *ip-address mask* [**secondary**]
5. **encapsulation ppp**
6. **dialer idle-timeout** *seconds* [**inbound** | **either**]
7. **dialer load-threshold** *load*
8. **dialer map** *protocol next-hop-address* [**name** *hostname*] [**spc**] [**speed 56** | **64**] [**broadcast**] [*dial-string*[**:***isdn-subaddress*]]
9. **dialer-group** *group-number*
10. **ppp authentication pap**
11. **ppp multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface bri** *number* **Example:** `Device(config)# interface bri 1`	Specifies an interface and benters interface configuration mode.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 192.0.2.0 255.255.255.224`	Provides an appropriate protocol address for the interface.
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 6	**dialer idle-timeout** *seconds* [**inbound** | **either**] **Example:** `Device(config-if)# dialer idle-timeout 60`	Specifies the duration of idle time in seconds after which a line will be disconnected. • By default, outbound traffic will reset the dialer idle timer. Adding the **either** keyword causes both inbound and outbound traffic to reset the timer; adding the **inbound** keyword causes only inbound traffic to reset the timer.
Step 7	**dialer load-threshold** *load* **Example:** `Device(config-if)# dialer load-threshold 60`	Specifies the dialer load threshold for bringing up additional WAN links.
Step 8	**dialer map** *protocol next-hop-address* [**name** *hostname*] [**spc**] [**speed 56** | **64**] [**broadcast**] [*dial-string*[**:***isdn-subaddress*]] **Example:** `Device(config-if)# dialer map protocol 192.0.2.1`	Configures the ISDN interface to call the remote site.
Step 9	**dialer-group** *group-number* **Example:** `Device(config-if)# dialer-group 3`	Controls access to this interface by adding it to a dialer access group.
Step 10	**ppp authentication pap** **Example:** `Device(config-if)# ppp authentication pap`	(Optional) Enables PPP authentication.
Step 11	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Configures MLP on the dialer rotary group.

Configuring MLP on Multiple ISDN BRI Interfaces

To enable MLP on multiple ISDN BRI interfaces, set up a dialer rotary interface and configure it for Multilink PPP, and then configure the BRI interfaces separately and add them to the same rotary group.

To set up the dialer rotary interface for the BRI interfaces, perform the following task:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *dialer number*
4. **ip address** *address mask*
5. **encapsulation ppp**
6. **dialer in-band**
7. **dialer idle-timeout** *seconds* [**inbound** | **either**]
8. **dialer map** *protocol next-hop-address* [**name** *hostname*] [**spc**] [**speed 56** | **64**] [**broadcast**] [*dial-string*[**:***isdn-subaddress*]]
9. **dialer rotary-group** *number*
10. **dialer load-threshold** *load*
11. **dialer-group** *number*
12. **ppp authentication chap**
13. **ppp multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface dialer** *number* **Example:** `Device(config)# interface dialer 1`	Specifies the dialer rotary interface and enters interface configuration mode.
Step 4	**ip address** *address mask* **Example:** `Device(config-if)# ip address 192.0.2.0 255.255.255.224`	Specifies the protocol address for the dialer rotary interface.

	Command or Action	Purpose
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 6	**dialer in-band** **Example:** `Device(config-if)# dialer in-band`	Specifies in-band dialing.
Step 7	**dialer idle-timeout** *seconds* [**inbound** | **either**] **Example:** `Device(config-if)# dialer idle-timeout 60`	Specifies the duration of idle time in seconds after which a line will be disconnected. • By default, outbound traffic will reset the dialer idle timer. Adding the **either** keyword causes both inbound and outbound traffic to reset the timer; adding the **inbound** keyword causes only inbound traffic to reset the timer.
Step 8	**dialer map** *protocol next-hop-address* [**name** *hostname*] [**spc**] [**speed 56** | **64**] [**broadcast**] [*dial-string*[**:***isdn-subaddress*]] **Example:** `Device(config-if)# dialer map protocol 192.0.2.1`	Maps the next hop protocol address and name to the dial string needed to reach it.
Step 9	**dialer rotary-group** *number* **Example:** `Device(config-if)# dialer rotary-group 1`	Adds the interface to the rotary group.
Step 10	**dialer load-threshold** *load* **Example:** `Device(config-if)# dialer load-threshold 2`	Specifies the dialer load threshold, using the same threshold as the individual BRI interfaces.
Step 11	**dialer-group** *number* **Example:** `Device(config-if)# dialer-group 2`	Controls access to the interface by adding it to a dialer access group.
Step 12	**ppp authentication chap** **Example:** `Device(config-if)# ppp authentication chap`	(Optional) Enables PPP CHAP authentication.
Step 13	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.

If you do not use PPP authentication procedures (Step 10), your telephone service must pass caller ID information.

Repeat Steps 1 through 9 for each BRI that you want to belong to the same dialer rotary group.

When MLP is configured and you want a multilink bundle to be connected indefinitely, use the **dialer idle-timeout** command to set a very high idle timer. The **dialer load-threshold 1** command does not keep a multilink bundle of *n* links connected indefinitely and the **dialer load-threshold 2** command does not keep a multilink bundle of two links connected indefinitely.)

Note Prior to Cisco IOS Release 12.1, when MLP was used in a dialer profile, a virtual access interface was always created as the bundle. It was bound to both the B channel and the dialer profile interfaces after creation and cloning. The dialer profile interface could act as the bundle without help from a virtual access interface. But with the Dynamic Multiple Encapsulations feature available in Cisco IOS Release 12.1, it is no longer the virtual access interface that is added into the connected group of the dialer profile, but the dialer profile itself. The dialer profile becomes a connected member of its own connected group. See the "Dynamic Multiple Encapsulations over ISDN Example" in the module "Configuring Peer-to-Peer DDR with Dialer Profiles" in this module, for more information about dynamic multiple encapsulations and its relation to Multilink PPP.

For an example of configuring MLP on multiple ISDN BRI interfaces, see the section "Example: MLP on Multiple ISDN BRI Interfaces" section on page 43.

Configuring MLP Using Multilink Group Interfaces

MLP can be configured by assigning a multilink group to a virtual template configuration. Virtual templates allow a virtual access interface to dynamically clone interface parameters from the specified virtual template. If a multilink group is assigned to a virtual template, and then the virtual template is assigned to a physical interface, all links that pass through the physical interface will belong to the same multilink bundle.

Note If a multilink group interface has one member link, the amount of bandwidth available will not change when a multilink interface is shut down. Therefore, you can shut down the multilink interface by removing its link.

A multilink group interface configuration will override a global multilink virtual template configured with the **multilink virtual template** command.

Multilink group interfaces can be used with ATM, PPP over Frame Relay, and serial interfaces.

To configure MLP using a multilink group interface, perform the following tasks:

- Configure the multilink group.
- Assign the multilink group to a virtual template.
- Configure the physical interface to use the virtual template.

Perform the following tasks in this section to configure the multilink group. For an example of how to configure MLP over an ATM PVC using a multilink group, see the section "Example: MLP Using Multilink Group Interfaces over ATM" section on page 44.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface multilink** *group-number*

4. **ip address** *address mask*
5. **encapsulation ppp**
6. **exit**
7. **interface virtual template** *number*
8. **ppp multilink group** *group-number*
9. **exit**
10. **interface atm** *interface-number.subinterface-number* **point-to-point**
11. **pvc** *vpi/vli*
12. **protocol ppp virtual-template** *name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *group-number* **Example:** `Device(config)# interface multilink 2`	Creates a multilink bundle and enters interface configuration mode to configure the bundle.
Step 4	**ip address** *address mask* **Example:** `Device(config-if)# ip address 192.0.2.1 255.255.255.224`	Sets a primary IP address for an interface.
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 7	**interface virtual template** *number* **Example:** `Device(config)# interface virtual template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode.

	Command or Action	Purpose
Step 8	**ppp multilink group** *group-number* **Example:** `Device(config-if)# ppp multilink group 2`	Restricts a physical link to joining only a designated multilink group interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 10	**interface atm** *interface-number.subinterface-number* **point-to-point** **Example:** `Device(config)# interface atm 1.2 point-to-point`	Configures an ATM interface and enters interface configuration mode.
Step 11	**pvc** *vpi/vci* **Example:** `Device(config-if)# pvc 1/100`	Creates or assigns a name to an ATM permanent virtual circuit (PVC), specifies the encapsulation type on an ATM PVC, and enters ATM virtual circuit configuration mode.
Step 12	**protocol ppp virtual-template** *name* **Example:** `Device(config-if-atm-vc)# protocol ppp virtual-template 2`	Configures VC multiplexed encapsulation on a PVC.
Step 13	**end** **Example:** `Device(config-if-atm-vc)# end`	Exits ATM virtual circuit configuration mode.

Changing the Default Endpoint Discriminator

By default, when the system negotiates use of MLP with the peer, the value that is supplied for the endpoint discriminator is the same as the username used for authentication. That username is configured for the interface by the Cisco IOS **ppp chap hostname** or **ppp pap sent-username** command, or defaults to the globally configured hostname (or stack group name, if this interface is a Stack Group Bidding Protocol, or SGBP, group member).

Perform this task to override or change the default endpoint discriminator. For an example of how to change the default endpoint discriminator, see the "Example: Changing the Default Endpoint Discriminator" section on page 37.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virutal template** *number*

4. **ppp multilink endpoint** {**hostname** | **ip** *ipaddress* | **mac** *LAN-interface* | **none** | **phone** *telephone-number* | **string char-string**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual template** *number* **Example:** `Device(config)# interface virtual template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces and enters interface configuration mode.
Step 4	**ppp multilink endpoint** {**hostname** | **ip** *ipaddress* | **mac** *LAN-interface* | **none** | **phone** *telephone-number* | **string** *char-string*} **Example:** `Device(config-if)# ppp multilink endpoint ip 192.0.2.0`	Overrides or changes the default endpoint discriminator the system uses when negotiating the use of MLP with the peer.

Configuring MLP Interleaving

Perform the following tasks to configure MLP and interleaving on a configured and operational interface or virtual interface template.

Configuring MLP Interleaving and Queueing

Interleaving on MLP allows large packets to be multilink encapsulated and fragmented into a small enough size to satisfy the delay requirements of real-time traffic; small real-time packets are not multilink encapsulated and are sent between fragments of the large packets. The interleaving feature also provides a special transmit queue for the smaller, delay-sensitive packets, enabling them to be sent earlier than other flows.

Weighted fair queueing on MLP works on the packet level, not at the level of multilink fragments. Thus, if a small real-time packet gets queued behind a larger best-effort packet and no special queue has been reserved for real-time packets, the small packet will be scheduled for transmission only after all the fragments of the larger packet are scheduled for transmission.

Weighted fair queueing is supported on all interfaces that support Multilink PPP, including MLP virtual access interfaces and virtual interface templates. Weighted fair queueing is enabled by default.

Fair queueing on MLP overcomes a prior restriction. Previously, fair queueing was not allowed on virtual access interfaces and virtual interface templates. Interleaving provides the delay bounds for delay-sensitive voice packets on a slow link that is used for other best-effort traffic.

Interleaving applies only to interfaces that can configure a multilink bundle interface. These restrictions include virtual templates, dialer interfaces, and ISDN BRI or PRI interfaces.

Multilink and fair queueing are not supported when a multilink bundle is off-loaded to a different system using Multichassis Multilink PPP (MMP). Thus, interleaving is not supported in MMP networking designs.

MLP support for interleaving can be configured on virtual templates, dialer interfaces, and ISDN BRI or PRI interfaces. To configure interleaving, complete the following tasks:

- Configure the dialer interface, BRI interface, PRI interface, or virtual template.
- Configure MLP and interleaving on the interface or template.

Note Fair queueing, which is enabled by default, must remain enabled on the interface.

Note Interleaving statistics can be displayed by using the **show interfaces** command, specifying the particular interface on which interleaving is enabled. Interleaving data is displayed only if there are interleaves. For example, the following line shows interleaves:

```
Output queue: 315/64/164974/31191 (size/threshold/drops/interleaves)
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual template** *number*
4. **ppp multilink**
5. **ppp multilink interleave**
6. **ppp multilink fragment delay** *milliseconds*
7. **ip rtp reserve** *lowest-udp-port range-of-ports* [*maximum-bandwidth*]
8. **exit**
9. **multilink virtual-template** *virtual-template-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface virtual template** *number* **Example:** `Device(config)# interface virtual template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode.
Step 4	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.
Step 5	**ppp multilink interleave** **Example:** `Device(config-if)# configure terminal`	Enables interleaving of packets among the fragments of larger packets on an MLP bundle.
Step 6	**ppp multilink fragment delay** *milliseconds* **Example:** `Device(config-if)# ppp multilink fragment delay 50`	Specifies a maximum size, in units of time, for packet fragments on an MLP bundle.
Step 7	**ip rtp reserve** *lowest-udp-port range-of-ports* [*maximum-bandwidth*] **Example:** `Device(config-if)# ip rtp reserve 1 2`	Reserves a special queue for real-time packet flows to specified destination UDP ports, allowing real-time traffic to have higher priority than other flows.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 9	**multilink virtual-template** *virtual-template-number* **Example:** `Device(config)# multilink virtual-template 1`	For virtual templates only, applies the virtual template to the multilink bundle. **Note** This step is not used for ISDN or dialer interfaces.

Configuring MLP Inverse Multiplexer and Distributed MLP

The distributed MLP (dMLP) feature combines T1/E1 lines in a WAN line card on a Cisco 7600 series router into a bundle that has the combined bandwidth of the multiple T1/E1 lines. You choose the number of bundles and the number of T1/E1 lines in each bundle, which allows you to increase the bandwidth of your network links beyond that of a single T1/E1 line without having to purchase a T3 line.

Nondistributed MLP is not supported on the Cisco 7600 series router. With distributed MLP, you can increase the router's total capacity.

The MLP Inverse Multiplexer feature was designed for Internet service providers (ISPs) that want to have the bandwidth of multiple T1 lines with performance comparable to that of an inverse multiplexer without the need of buying standalone inverse-multiplexing equipment. A Cisco router supporting dMLP can bundle multiple T1 lines in a CT3 or CE3 interface or channelized STM1. Bundling is more economical than purchasing an inverse multiplexer, and eliminates the need to configure another piece of equipment.

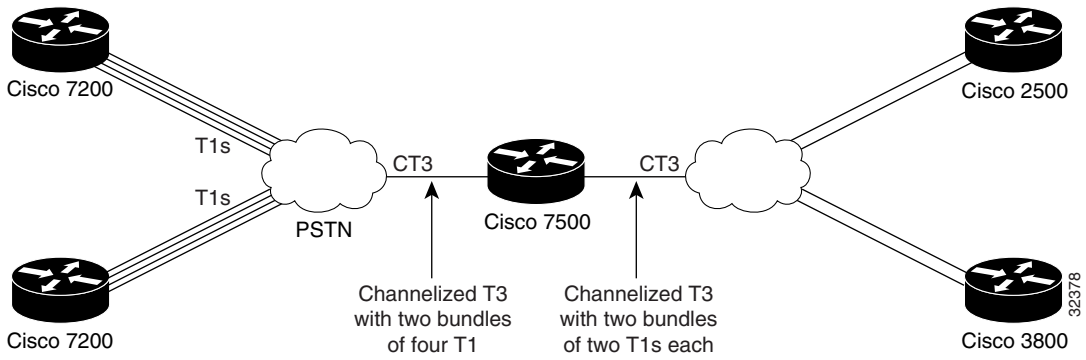
This feature supports the CT3 CE3 data rates without taxing the Route Processor (RP) and CPU by moving the data path to the line card. This feature also allows remote sites to purchase multiple T1 lines instead of a T3 line, which is especially useful when the remote site does not need the bandwidth of an entire T3 line.

This feature allows multilink fragmentation to be disabled, so multilink packets are sent using Cisco Express Forwarding on all platforms, if fragmentation is disabled. Cisco Express Forwarding is supported with fragmentation enabled or disabled.

Note If a router cannot send out all the packets (some packets are dropped by Quality of Service (QoS)), late drops occur. These late drops are displayed when the **show interface** command is executed.
If there is no service policy on the dMLP interface, when a **ppp multilink interleave** is configured on the dMLPPP interface, a QoS policy is enabled internally.

Figure 2 shows a typical network using a dMLP link. The Cisco 7600 series router is connected to the network with a CT3 line that has been configured with dMLPP to carry two bundles of four T1 lines each. One of these bundles goes out to a Cisco 2500 series router and the other goes out to a Cisco 3800 series router.

Figure 2 Diagram of a Typical VIP MLP Topology



Before beginning the MLP Inverse Multiplexer configuration tasks, make note of the following prerequisites and restrictions.

Prerequisites

- Distributed Cisco Express Forwarding switching must be enabled for distributed MLP.
- One of the following port adapters is required:
  - CT3IP
  - PA-MC-T3
  - PA-MC-2T3+
  - PA-MC-E3
  - PA-MC-8T1
  - PA-MC-4T1
  - PA-MC-8E1
- All 16 E1s can be bundled from a PA-MC-E3 in a VIP4-80.

Restrictions

The following restrictions apply to the dMLP feature:

Note Distributed MLP is supported only for member links configured at T1/E1 or subrate T1/E1 speeds. Channelized STM-1/T3/T1 interfaces also support dMLP at T1/E1 or subrate T1/E1 speeds. Distributed MLP is not supported for member links configured at clear-channel T3/E3 or higher interface speeds.

- T1 and E1 lines cannot be mixed in a bundle.
- T1 lines in a bundle should have the same bandwidth.
- All lines in a bundle must reside on the same port adapter.
- MLP bundles across FlexWAN or Enhanced FlexWAN port adapters are not supported.
- Hardware compression is not supported.
- Encryption is not supported.
- Software compression is not recommended because CPU usage would void performance gains.
- The maximum differential delay supported is 50 milliseconds (ms).
- Fragmentation is not supported on the transmit side.
- dMLP across shared port adapters (SPAs) is not supported.
- Hardware and software compression is not supported.
- Encryption is not supported.
- The maximum differential delay supported is 50 ms when supported in hardware, and 100 ms when supported in software.

Enabling fragmentation reduces the delay latency among bundle links, but adds some load to the CPU. Disabling fragmentation may result in better throughput.

If your data traffic is consistently of a similar size, we recommend disabling fragmentation. In this case, the benefits of fragmentation may be outweighed by the added load on the CPU.

To configure a multilink bundle, perform the tasks in the following sections:

- Creating a Multilink Bundle, page 32 (required)
- Assigning an Interface to a Multilink Bundle, page 33 (required)
- Disabling PPP Multilink Fragmentation, page 35 (optional)

Creating a Multilink Bundle

Perform the following tasks to create a multilink bundle.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface multilink** *group-number*
4. **ip address** *address mask*
5. **encapsulation ppp**
6. **ppp multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *group-number* **Example:** `Device(config)# interface multilink 10`	Assigns a multilink group number and enters interface configuration mode.
Step 4	**ip address** *address mask* **Example:** `Device(config-if)# ip address 192.0.2.9 255.255.255.224`	Assigns an IP address to the multilink interface.
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 6	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.

Assigning an Interface to a Multilink Bundle

Perform this task to assign an interface to a multilink bundle.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface multilink** *group number*
4. **no ip address**
5. **keepalive**
6. **encapsulation ppp**
7. **ppp multilink group** *group-number*
8. **ppp multilink**
9. **ppp authentication chap**
10. **pulse-time seconds**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *group-number* **Example:** `Device(config)# interface multilink 10`	Assigns a multilink group number and enters interface configuration mode.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Removes any specified IP address.
Step 5	**keepalive** **Example:** `Device(config-if)# keepalive`	Sets the frequency of keepalive packets.
Step 6	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation.
Step 7	**ppp multilink group** *group-number* **Example:** `Device(config-if)# ppp multilink 12`	Restricts a physical link to joining only the designated multilink-group interface.
Step 8	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables Multilink PPP.
Step 9	**ppp authentication chap** **Example:** `Device(config-if)# ppp authentication chap`	(Optional) Enables CHAP authentication.
Step 10	**pulse-time** *seconds* **Example:** `Device(config-if)# pulse-time 10`	(Optional) Configures DTR signal pulsing.

Caution Do not install a router to the peer address while configuring an MLP lease line. This installation can be disabled when **no ppp peer-neighbor-route** command is used under the MLPPP bundle interface.

Disabling PPP Multilink Fragmentation

Perform the following task to disable PPP multilink fragmentation.

SUMMARY STEPS

1. **enable**
2. **configuration terminal**
3. **interface multilink** *group number*
4. **ppp multilink fragment disable**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *group-number* **Example:** `Device(config)# interface multilink 10`	Assigns a multilink group number and enters interface configuration mode.

Step 4	**ppp multilink fragment disable**
	
	Example:
	`Device(config-if)#`
	`ppp multilink fragment disable`
		(Optional) Disables PPP multilink fragmentation.

Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits privileged EXEC mode.

Monitoring and Maintaining PPP and MLP Interfaces

Perform this task to display MLP and MMP bundle information.

SUMMARY STEPS

1. **enable**
2. **show ppp multilink**
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ppp multilink** **Example:** `Device# show ppp multilink`	Displays MLP and MMP bundle information.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode.

Configuration Examples for PPP and MLP

The following sections provide various PPP configuration examples:

Examples: CHAP with an Encrypted Password:

The following examples show how to enable CHAP on serial interface 0 of three devices:

Configuration of Device yyy

```
hostname yyy
interface serial 0
```

```
 encapsulation ppp
 ppp authentication chap
username xxx password secretxy
username zzz password secretzy
```

Configuration of Device xxx

```
hostname xxx
interface serial 0
 encapsulation ppp
 ppp authentication chap
username yyy password secretxy
username zzz password secretxz
```

Configuration of Device zzz

```
hostname zzz
interface serial 0
 encapsulation ppp
 ppp authentication chap
username xxx password secretxz
username yyy password secretzy
```

When you look at the configuration file, the passwords are encrypted and the display looks similar to the following:

```
hostname xxx
interface serial 0
 encapsulation ppp
 ppp authentication chap
username yyy password 7 121F0A18
username zzz password 7 1329A055
```

Example: Changing the Default Endpoint Discriminator

The following partial example changes the MLP endpoint discriminator from the default CHAP hostname C-host1 to the E.164-compliant telephone number 555-0100:

```
.
.
.
interface dialer 0
 ip address 10.1.1.4 255.255.255.0
 encapsulation ppp
 dialer remote-name R-host1
 dialer string 23456
 dialer pool 1
 dialer-group 1
 ppp chap hostname C-host1
 ppp multilink endpoint phone 555-0100
.
.
.
```

Example: DHCP Network Control

The following partial example shows how to add the **ip dhcp-client network-discovery** command to the "Example: IP Address Pooling" section on page 38 to allow peer routers to more dynamically discover DNS and NetBIOS name servers. If the **ip dhcp-client network-discovery** command is disabled, the system falls back to the static configurations made using the **async-bootp dns-server** and **async-bootp nb-server** global configuration commands.

```
!
hostname secret
!
aaa new-model
aaa authentication login default local
aaa authentication ppp default if-needed local
aaa authentication ppp chap local
enable secret 5 encrypted-secret
enable password EPassWd1
!
username User1 password 0 PassWd2
username User2 password 0 PassWd3
username User3 password 0 PassWd4
no ip domain-lookup
ip dhcp-server 10.47.0.131
ip dhcp-client network-discovery informs 2 discovers 2 period 12
async-bootp gateway 10.47.0.1
async-bootp nbns-server 10.47.0.131
isdn switch-type primary-4ess
.
.
.
```

Example: IP Address Pooling

The following example shows how to configure a modem to dial in to a Cisco access server and obtain an IP address from the DHCP server. This configuration allows the user to log in and browse an NT network. Notice that the dialer 1 and group-async 1 interfaces are configured with the **ip unnumbered loopback** command, so that the broadcast can find the dialup clients and the client can see the NT network.

```
!
hostname secret
!
aaa new-model
aaa authentication login default local
aaa authentication ppp default if-needed local
aaa authentication ppp chap local
enable secret 5 encrypted-secret
enable password EPassWd1
!
username User1 password 0 PassWd2
username User2 password 0 PassWd3
username User3 password 0 PassWd4
no ip domain-lookup
ip dhcp-server 10.47.0.131
async-bootp gateway 10.47.0.1
async-bootp nbns-server 10.47.0.131
isdn switch-type primary-4ess
!
!
```

```
controller t1 0
 framing esf
 clock source line primary
 linecode b8zs
 pri-group timeslots 1-24
!
controller t1 1
 framing esf
 clock source line secondary
 linecode b8zs
!
interface loopback 0
 ip address 10.47.252.254 255.255.252.0
!
interface ethernet 0
 ip address 10.47.0.5 255.255.252.0
 ip helper-address 10.47.0.131
 ip helper-address 10.47.0.255
 no ip route-cache
 no ip mroute-cache
!
interface serial 0
 no ip address
 no ip mroute-cache
 shutdown
!
interface serial 1
 no ip address
 shutdown
!
interface serial 0:23
 no ip address
 encapsulation ppp
 no ip mroute-cache
 dialer rotary-group 1
 dialer-group 1
 isdn incoming-voice modem
 no fair-queue
 no cdp enable
!
interface group-async 1
 ip unnumbered loopback 0
 ip helper-address 10.47.0.131
 ip tcp header-compression passive
 encapsulation ppp
 no ip route-cache
 no ip mroute-cache
 async mode interactive
 peer default ip address dhcp
 no fair-queue
 no cdp enable
 ppp authentication chap
 group-range 1 24
!
interface dialer 1
 ip unnumbered loopback 0
 encapsulation ppp
 dialer in-band
 dialer-group 1
 no peer default ip address
 no fair-queue
 no cdp enable
 ppp authentication chap
 ppp multilink
```

```
!
router ospf 172
 redistribute connected subnets
 redistribute static
 network 10.47.0.0 0.0.3.255 area 0
 network 10.47.156.0 0.0.3.255 area 0
 network 10.47.168.0 0.0.3.255 area 0
 network 10.47.252.0 0.0.3.255 area 0
!
ip local pool RemotePool 10.47.252.1 10.47.252.24
ip classless
ip route 10.0.140.0 255.255.255.0 10.59.254.254
ip route 10.2.140.0 255.255.255.0 10.59.254.254
ip route 10.40.0.0 255.255.0.0 10.59.254.254
ip route 10.59.254.0 255.255.255.0 10.59.254.254
ip route 172.23.0.0 255.255.0.0 10.59.254.254
ip route 192.168.0.0 255.255.0.0 10.59.254.254
ip ospf name-lookup
no logging buffered
access-list 101 deny ip any host 255.255.255.255
access-list 101 deny ospf any any
access-list 101 permit ip any any
dialer-list 1 protocol ip list 101
snmp-server community public RO
!
line con 0
line 1 24
 autoselect during-login
 autoselect ppp
 modem InOut
 transport input all
line aux 0
line vty 0 4
 password Password
!
scheduler interval 100
end
```

Example: MPPC Interface Configuration

The following example shows how to configure asynchronous interface 1 to implement MPPC and ignore the protocol field compression flag negotiated by LCP:

```
interface async1
 ip unnumbered ethernet0
 encapsulation ppp
 async default routing
 async dynamic routing
 async mode interactive
 peer default ip address 172.21.71.74
 compress mppc ignore-pfc
```

The following example creates a virtual access interface (virtual template interface 1) and serial interface 0, which is configured for X.25 encapsulation. MPPC values are configured on the virtual template interface and will ignore the negotiated protocol field compression flag.

```
interface ethernet0
 ip address 172.20.30.102 255.255.255.0
!
interface virtual-template1
 ip unnumbered ethernet0
```

```
 peer default ip address pool vtemp1
 compress mppc ignore-pfc
!
interface serial0
 no ipaddress
no ip mroute-cache
encapsulation x25
x25 win 7
x25 winout 7
x25 ips 512
x25 ops 512
clock rate 50000
!
ip local pool vtemp1 172.20.30.103 172.20.30.104
ip route 0.0.0.0 0.0.0.0 172.20.30.1
!
translate x25 31320000000000 virtual-template 1
```

Examples: MLP

This section contains the following MLP examples:

- Example: MLP on Synchronous Serial Interfaces, page 41
- Example: MLP on One ISDN BRI Interface, page 43
- Example: MLP on Multiple ISDN BRI Interfaces, page 43
- Example: MLP Inverse Multiplexer Configuration, page 44
- Example: MLP Using Multilink Group Interfaces over ATM, page 44
- Example: Changing the Default Endpoint Discriminator, page 37

Example: MLP on Synchronous Serial Interfaces

The following example shows how the configuration commands are used to create the inverse multiplexing application:

Device A Configuration

```
hostname DeviceA
!
!
username DeviceB password your_password
ip subnet-zero
multilink virtual-template 1
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 ppp authentication chap
 ppp multilink
!
interface Serial0
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial1
 no ip address
```

```
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial2
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial3
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Ethernet0
 ip address 10.17.1.254 255.255.255.0
!
router rip
network 10.0.0.0
!
end
```

Device B Configuration

```
hostname DeviceB
!
!
username DeviceB password your_password
ip subnet-zero
multilink virtual-template 1
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 ppp authentication chap
 ppp multilink
!
interface Serial0
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial1
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial2
 no ip address
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Serial3
 no ip address
```

```
 encapsulation ppp
 no fair-queue
 ppp multilink
 pulse-time 3
!
interface Ethernet0
 ip address 10.17.2.254 255.255.255.0
!
router rip
network 10.0.0.0
!
end
```

Example: MLP on One ISDN BRI Interface

The following example shows how to enable MLP on BRI interface 0. When a BRI is configured, no dialer rotary group configuration is required, because an ISDN interface is a rotary group by default.

```
interface bri 0
 description connected to ntt 81012345678902
 ip address 172.31.1.7 255.255.255.0
 encapsulation ppp
 dialer idle-timeout 30
 dialer load-threshold 40 either
 dialer map ip 172.31.1.8 name user1 81012345678901
 dialer-group 1
 ppp authentication pap
 ppp multilink
```

Example: MLP on Multiple ISDN BRI Interfaces

The following example shows how to configure multiple ISDN BRI interfaces to belong to the same dialer rotary group for Multilink PPP. The **dialer rotary-group** command is used to assign each of the ISDN BRI interfaces to that dialer rotary group.

```
interface BRI 0
 no ip address
 encapsulation ppp
 dialer idle-timeout 500
 dialer rotary-group 0
 dialer load-threshold 30 either
!
interface BRI 1
 no ip address
 encapsulation ppp
 dialer idle-timeout 500
 dialer rotary-group 0
 dialer load-threshold 30 either
!
interface BRI 2
 no ip address
 encapsulation ppp
 dialer idle-timeout 500
 dialer rotary-group 0
 dialer load-threshold 30 either
!
interface Dialer 0
 ip address 10.0.0.2 255.0.0.0
 encapsulation ppp
 dialer in-band
 dialer idle-timeout 500
```

```
 dialer map ip 10.0.0.1 name user1 broadcast 81012345678901
 dialer load-threshold 30 either
 dialer-group 1
 ppp authentication chap
 ppp multilink
```

Example: MLP Using Multilink Group Interfaces over ATM

The following example shows how to configure MLP over an ATM PVC using a multilink group:

```
interface multilink 1
 ip address 10.200.83.106 255.255.255.252
 ip tcp header-compression iphc-format delay 20000
 service policy output xyz
 encapsulation ppp
 exit
 ppp multilink
 ppp multilink fragment delay 10
 ppp multilink interleave
 ppp timeout multilink link remove 10
 ip rtp header-compression iphc-format

interface virtual-template 3
 bandwidth 128
 ppp multilink group 1

interface atm 4/0.1 point-to-point
 pvc 0/32
 abr 100 80
 protocol ppp virtual-template 3
```

Example: MLP Inverse Multiplexer Configuration

This example shows how to verify the display information of the newly created multilink bundle:

```
Device# show ppp multilink

Multilink1, bundle name is group1
Bundle is Distributed
0 lost fragments, 0 reordered, 0 unassigned, sequence 0x0/0x0 rcvd/sent
0 discarded, 0 lost received, 1/255 load
Member links:4 active, 0 inactive (max not set, min not set)
 Serial1/0/0:1
 Serial1/0/0/:2
 Serial1/0/0/:3
 Serial1/0/0/:4
```

Example: MLP Interleaving and Queueing for Real-Time Traffic

The following example defines a virtual interface template that enables MLP interleaving and a maximum real-time traffic delay of 20 milliseconds, and then applies that virtual template to the MLP bundle:

```
interface virtual-template 1
 ip unnumbered ethernet 0
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment delay 20
 ip rtp interleave 32768 20 1000
multilink virtual-template 1
```

The following example enables MLP interleaving on a dialer interface that controls a rotary group of BRI interfaces. This configuration permits IP packets to trigger calls.

```
interface BRI 0
 description connected into a rotary group
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 1
 no ip address
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 2
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 3
 no ip address
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 4
 encapsulation ppp
 dialer rotary-group 1
!
interface Dialer 0
 description Dialer group controlling the BRIs
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
 dialer map ip 10.1.1.2 name name1 14802616900
 dialer-group 1
 ppp authentication chap
! Enables Multilink PPP interleaving on the dialer interface and reserves
! a special queue.
 ppp multilink
 ppp multilink interleave
 ip rtp reserve 32768 20 1000
! Keeps fragments of large packets small enough to ensure delay of 20 ms or less.
 ppp multilink fragment delay 20
dialer-list 1 protocol ip permit
```

Example: Multilink Interface Configuration for Distributed MLP

In the following example, four multilink interfaces are created with distributed Cisco Express Forwarding switching and MLP enabled. Each of the newly created interfaces is added to a multilink bundle.

```
interface multilink1
 ip address 10.0.0.0 10.255.255.255
 ppp chap hosstname group 1
 ppp multilink
 ppp multilink group 1

interface serial 1/0/0:1
 no ip address
 encapsulation ppp
 ip route-cache distributed
 no keepalive
 ppp multilink
 ppp multilink group 1
```

```
interface serial 1/0/0:2
 no ip address
 encapsulation ppp
 ip route-cache distributed
 no keepalive
 ppp chap hostname group 1
 ppp multilink
 ppp multilink group 1

interface serial 1/0/0:3
 no ip address
 encapsulation ppp
 ip route-cache distributed
 no keepalive
 ppp chap hostname group 1
 ppp multilink
 ppp multilink group 1

interface serial 1/0/0:4
 no ip address
 encapsulation ppp
 ip route-cache distributed
 no keepalive
 ppp chap hostname group 1
 ppp multilink
 ppp multilink group 1
```

Example: PAP commands for a one way authentication

The following example shows how to authenticate PAP commands for a one way authentication scenario:

Note Only the relevant sections of the configuration are shown.

```
Calling Side (Client) Configuration
interface BRI0

! --- BRI interface for the dialout.

 ip address negotiated
 encapsulation ppp

! --- Use PPP encapsulation. This command is a required for PAP.

 dialer string 3785555 class 56k

! --- Number to dial for the outgoing connection.

 dialer-group 1
 isdn switch-type basic-ni
 isdn spid1 51299611110101 9961111
 isdn spid2 51299622220101 9962222
 ppp authentication pap callin

! --- Use PAP authentication for incoming calls.
! --- The callin keyword has made this a one-way authentication scenario.
! --- This router (client) will not request that the peer (server) authenticate
! --- itself back to the client.
```

```
 ppp pap sent-username PAPUSER password 7 <deleted>

! --- Permit outbound authentication of this router (client) to the peer.
! --- Send a PAP AUTH-REQ packet to the peer with the username PAPUSER and password.
! --- The peer must have the username PAPUSER and password configured on it.

Receiving Side (Server) Configuration
username PAPUSER password 0 cisco

! --- Username PAPUSER is the same as the one sent by the client.
! --- Upon receiving the AUTH-REQ packet from the client, we will verify that the
! --- username and password match the one configured here.

interface Serial0:23

! --- This is the D-channel for the PRI on the access server receiving the call.

 ip unnumbered Ethernet0
 no ip directed-broadcast
 encapsulation ppp

! --- Use PPP encapsulation. This command is a required for PAP.

 dialer-group 1
 isdn switch-type primary-ni
 isdn incoming-voice modem
 peer default ip address pool default
 fair-queue 64 256 0
 ppp authentication pap

! --- Use PAP authentication for incoming calls.
! --- This router (server) will request that the peer authenticate itself to us.
! --- Note: the callin option is not used as this router is not initiating the call.
```

Example: T3 Controller Configuration for an MLP Multilink Inverse Multiplexer

The following example shows how to configure the T3 controller and create four channelized interfaces:

```
controller T3 1/0/0
framing m23
cablelength 10
t1 1 timeslots 1-24
t1 2 timeslots 1-24
t1 3 timeslots 1-24
t1 4 timeslots 1-24
```

Example: User Maximum Links Configuration

The following example shows how to configure the username user1 and establish a maximum of five connections. user1 can connect through serial interface 1/0, which has a dialer map configured for it, or through PRI interface 0/0:23, which has dialer profile interface 0 dedicated to it.

The **aaa authorization network default local** command must be configured. PPP encapsulation and authentication must be enabled on all the interfaces that user1 can connect to.

```
aaa new-model
aaa authorization network default local
enable secret password1
```

```
enable password password2
!
username user1 user-maxlinks 5 password password3
!
interface Serial0/0:23
 no ip address
 encapsulation ppp
 dialer pool-member 1
 ppp authentication chap
 ppp multilink
!
interface Serial1/0
 ip address 209.165.201.1 255.255.255.0
 encapsulation ppp
 dialer in-band
 dialer map ip 10.2.2.13 name user1 12345
 dialer-group 1
 ppp authentication chap
!
interface Dialer0
 ip address 209.165.200.225 255.255.255.0
 encapsulation ppp
 dialer remote-name user1
 dialer string 23456
 dialer pool 1
 dialer-group 1
 ppp authentication chap
 ppp multilink
!
dialer-list 1 protocol ip permit
```

Additional References

The following sections provide references related to the Configuring Media-Independent PPP and Multilink PPP feature.

Related Documents

Related Topic	Document Title
Asynchronous SLIP and PPP	*"Configuring Asynchronous SLIP and PPP"* module in the *Cisco IOS Dial Technologies Configuration Guide*
MCHAP	*MS-CHAP Support*
Dial commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Dial Technologies Command Reference.*

RFCs

RFC	Title
RFC 1994	PPP Challenge Handshake Authentication Protocol (CHAP)

Technical Assistance

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

Feature Information for Configuring Media-Independent PPP and Multilink PPP

Table 1 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.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 1 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 1 Feature Information for Configuring Media-Independent PPP and Multilink PPP

Feature Name	Releases	Feature Information
Multilink PPP	11.2(1) 12.2(8)T 11.2(6)P 12.1(3)T 12.3(13)BC 12.2(27)SBB 12.2(31)SB2 15.0(1)M 12.2(33)SRE 15.2(2)S Cisco IOS Release 3.14S	Multilink PPP provides a method for spreading traffic across multiple physical WAN links. The following sections provide information about this feature: • Information About Media-Independent PPP and Multilink PPP, page 2 • How to Configure Media-Independent PPP and Multilink PPP, page 6 The following commands were introduced or modified: **ppp multilink**, **ppp multilink group**.

PPP/MLP MRRU Negotiation Configuration

First Published: March 1, 2004
Last Updated: November 20, 2014

The PPP/MLP MRRU Negotiation Configuration feature allows a device to send and receive frames over Multilink PPP (MLP) bundles that are larger than the default Maximum Receive Reconstructed Unit (MRRU) limit of 1524 bytes.

Finding Feature Information in This Module

Your Cisco IOS software release may not support all of the features documented in this module. For the latest feature information and caveats, see the release notes for your Cisco IOS software release. 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 PPP/MLP MRRU Negotiation Configuration" section on page 64.

Finding Support Information for Platforms and Cisco IOS and Catalyst OS Software Images

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

Contents

Prerequisites for PPP/MLP MRRU Negotiation Configuration

Before performing the tasks to configure the PPP/MLP MRRU Negotiation Configuration feature, you need to understand how to configure PPP and MLP. It will also be useful to be familiar with concepts presented in RFC 1990. See the "Related Documents" and "RFCs" sections for pointers to this information.

Restrictions for PPP/MLP MRRU Negotiation Configuration

This feature and its new interface configuration command are valid only on interfaces that support MLP.

Note Be careful when configuring MLP MRRU negotiation in a virtual private dialup network (VPDN) environment with an L2TP network server (LNS) that is not running Cisco IOS Release 12.3(7)T. The software performs strict matching on the MRRU values in earlier versions of the Cisco IOS software.

Information About PPP/MLP MRRU Negotiation Configuration

To configure PPP/MLP MRRU Negotiation Configuration, you need to understand the following concepts:

MRRU Negotiation on MLP

Before Cisco IOS Release 12.3(7)T, configuring the MRRU option negotiated on a multilink bundle with the MLP was not possible. Cisco IOS software always advertised an MRRU default value of 1524 bytes, which meant that the maximum transmission unit (MTU) of the peer's bundle interface was restricted to a value of 1524 bytes or fewer if the data transfer was to be successful. Users who wanted to benefit from MLP features had to accept limits on the MTU byte size setting.

The PPP/MLP MRRU Negotiation Configuration feature allows configuration control over MRRU negotiation. A new interface configuration command introduced with this feature, **ppp multilink mrru,** allows configuring the specific MRRU value that the device will advertise, and optionally establishing a lower boundary on the MRRU value of the peer.

MLP is a method for spreading traffic across multiple physical WAN links while providing packet fragmentation and reassembly, proper sequencing, multi-vendor interoperability, and load balancing on inbound and outbound traffic. MLP was developed to use the multiple bearer channels in ISDN, but is equally applicable to any situation in which multiple PPP links connect two systems, including asynchronous links.

When MLP is used, several physical interfaces can constitute one logical connection to the peer. To represent the logical connection, software provides a logical interface, often called the *bundle* interface. This interface will have the IP address, for instance, and the MTU setting of the interface that IP uses when it is deciding whether to fragment an IP datagram that needs to be forwarded. The physical interfaces simply forward individual MLP fragments or frames that are given to them by the bundle interface.

The result of having to decide whether to fragment a packet is that, whereas with simple PPP the interface MTU must not exceed the peer's MRRU, with MLP the MTU size of the bundle interface must not exceed the MRRU setting of the peer.

The MRRU settings on both sides need not be equal, but the "must not exceed" rule just specified must be followed; otherwise a system might send several fragments that, when reconstructed as a frame, will be too large for the peer's receive buffer.

Advertisement of a Specific MRRU Value

Where a PPP link is destined to join an existing MLP bundle, the MRRU value advertised on that link will be the MRRU of the existing bundle, in configurations where the software can determine which bundle the link is destined to join at the time the Link Control Protocol (LCP) is negotiated.

In Cisco IOS Release 12.0(28)S, this is the case for multilink groups only.

In Cisco IOS Release 12.3(7)T and later releases, this is the case for both multilink groups and dialer profiles that have already been bound to the physical interface at the time LCP is negotiated.

In all other cases, the MRRU value advertised on a link is by an order of preference, as follows:

- The value configured on the link interface with the **ppp multilink mrru** interface command, or the value inherited from the configuration of the **ppp multilink mrru** command on the parent interface. If both values are present, the link interface value has precedence.
- The value of the bundle interface MTU, if the bundle interface is known at the time LCP is negotiated. In Cisco IOS Release 12.0(28)S, this is the case for multilink groups only.
- The default MRRU value of 1524 bytes.

Peer MRRU Negotiation

By default, any peer MRRU value that is set higher thatn the lower boundary of what is considered to be acceptable will be acknowledged.

In addition, the **ppp multilink mrru** interface command will allow specifying a minimum required peer MRRU value. If a lower value has been configured on a link interface or is inherited from a parent interface, software will send a negative acknowledgment along with the required minimum MRRU value to any peer with MRRU values that are below the established threshold.

When the bundle interface comes up, its MTU will be reduced if it exceeds the peer's MRRU for the duration of the existence of the bundle.

When a link joins a bundle, it must have the same values configured for the local and remote MRRU as the bundle does. If not, the link will be dropped and an error message will be displayed.

How to Configure PPP/MLP MRRU Negotiation Configuration

The following sections describe how to configure the PPP/MLP MRRU Negotiation Configuration feature.

- Configuring PPP/MLP MRRU Negotiation Configuration on Virtual Templates, page 54 (required for virtual templates)
- Configuring PPP/MLP MRRU Negotiation Configuration on Multilink Groups, page 56 (required for multilink groups)
- Configuring PPP/MLP MRRU Negotiation Configuration on Dialer Interfaces, page 59 (required for dialer interfaces)

Configuring PPP/MLP MRRU Negotiation Configuration on Virtual Templates

In this task, you configure MRRU negotiation on the serial interface. The bundle interface will be a virtual access interface cloned from the virtual template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **multilink virtual-template** *number*
4. **interface virtual-template** *number*
5. **ip address** *ip-address mask*
6. **mtu** *bytes*
7. **exit**
8. **interface serial** *slot/port*
9. **ppp multilink**
10. **ppp multilink mrru** [**local** | **remote**] *mrru-value*
11. **mtu** *bytes*
12. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**multilink virtual-template** *number* **Example:** `Device(config)# multilink virtual-template 1`	Specifies a virtual template from which the specified MLP bundle interface can clone its interface parameters.
Step 4	**interface virtual-template** *number* **Example:** `Device(config)# interface virtual-template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.13.1.1 255.255.255.0`	Sets the IP address for the interface.
Step 6	**mtu** *bytes* **Example:** `Device(config-if)# mtu 1600`	(Optional) Adjusts the maximum packet size or MTU size. • Once you configure the MRRU on the bundle interface, you enable the DeviceDevice to receive large reconstructed MLP frames. You may want to configure the bundle MTU so the Device can transmit large MLP frames, although it is not strictly necessary. • The maximum recommended value for the bundle MTU is the value of the peer's MRRU. The default MTU for serial interfaces is 1500. The software will automatically reduce the bundle interface MTU if necessary, to avoid violating the peer's MRRU.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface serial** *slot/port* **Example:** `Device(config)# interface serial 0/0`	Selects a serial interface to configure and starts interface configuration mode.
Step 9	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables MLP on an interface.

	Command or Action	Purpose
Step 10	**ppp multilink mrru** [**local** | **remote**] *mrru-value* **Example:** `Device(config-if)# ppp multilink mrru local 1600`	Configures the MRRU value negotiated on a multilink bundle when MLP is used. • **local**—(Optional) Configures the local MRRU value. The default values for the local MRRU are the value of the multilink group interface MTU for multilink group members, and 1524 bytes for all other interfaces. • **remote**—(Optional) Configures the minimum value that the software will accept from the peer when it advertises its MRRU. By default, the software accepts any peer MRRU value of 128 or higher. You can specify a higher minimum acceptable MRRU value in a range from 128 to 16384 bytes.
Step 11	**mtu** *bytes* **Example:** `Device(config-if)# mtu 1600`	(Optional) Adjusts the maximum packet size or MTU size. • The default MTU for serial interfaces is 1500. • When the bundle interface MTU is tuned to a higher number, then depending upon the fragmentation configuration, the link interface may be given larger frames to transmit. • You must ensure that fragmentation is performed such that fragments are sized less than the link interface MTU (refer to command pages for the **ppp multilink fragmentation** and **ppp multilink fragment-delay** commands for more information about packet fragments), or configure the MTUs of the link interfaces such that they can transmit the larger frames.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode. • Return to Step 8 and configure additional interfaces, if necessary.

Troubleshooting Tips

Use the **debug ppp negotiation** command to verify and troubleshoot MRRU negotiation on virtual templates. Use the **show interface** command to verify MRRU negotiation on the interfaces.

Configuring PPP/MLP MRRU Negotiation Configuration on Multilink Groups

In this task, you configure MRRU negotiation on the multilink interface. The bundle interface is static, that is, always available.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface multilink** *number*

4. **ip address** *ip-address mask*
5. **ppp multilink mrru** [**local** | **remote**] *mrru-value*
6. **mtu** *bytes*
7. **exit**
8. **interface serial** *slot*/*port*
9. **ppp multilink**
10. **ppp multilink group** *group-number*
11. **mtu** *bytes*
12. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *number* **Example:** `Device(config)# interface multilink 10`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.13.1.1 255.255.255.0`	Sets the IP address for the interface.
Step 5	**ppp multilink mrru** [**local** | **remote**] *mrru-value* **Example:** `Device(config-if)# ppp multilink mrru local 1600`	Configures the MRRU value negotiated on a multilink bundle when MLP is used. • **local**—(Optional) Configures the local MRRU value. The default values for the local MRRU are the value of the multilink group interface MTU for multilink group members, and 1524 bytes for all other interfaces. • **remote**—(Optional) Configures the minimum value that the software will accept from the peer when it advertises its MRRU. By default, the software accepts any peer MRRU value of 128 or higher. You can specify a higher minimum acceptable MRRU value in a range from 128 to 16384 bytes.

	Command or Action	Purpose
Step 6	**mtu** *bytes* **Example:** `Device(config-if)# mtu 1600`	(Optional) Adjusts the maximum packet size or MTU size. • Once you configure the MRRU on the bundle interface, you enable the Device to receive large reconstructed MLP frames. You may want to configure the bundle MTU so the Device can transmit large MLP frames, although it is not strictly necessary. • The maximum recommended value for the bundle MTU is the value of the peer's MRRU. The default MTU for serial interfaces is 1500. The software will automatically reduce the bundle interface MTU if necessary, to avoid violating the peer's MRRU.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface serial** *slot/port* **Example:** `Device(config)# interface serial 0/0`	Selects a serial interface to configure and enters interface configuration mode.
Step 9	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables MLP on the interface.
Step 10	**ppp multilink group** *group-number* **Example:** `Device(config-if)# ppp multilink group 1`	Restricts a physical link to joining only a designated multilink-group interface.
Step 11	**mtu** *bytes* **Example:** `Device(config-if)# mtu 1600`	(Optional) Adjusts the maximum packet size or MTU size. • The default MTU for serial interfaces is 1500. • When the bundle interface MTU is tuned to a higher number, then depending upon the fragmentation configuration, the link interface may be given larger frames to transmit. • You must ensure that fragmentation is performed such that fragments are sized less than the link interface MTU (refer to command pages for the **ppp multilink fragmentation** and **ppp multilink fragment-delay** commands for more information about packet fragments), or configure the MTUs of the link interfaces such that they can transmit the larger frames.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

Troubleshooting Tips

Use the **debug ppp negotiation** command to verify and troubleshoot MRRU negotiation on multilink groups. Use the **show interface** command to verify MRRU negotiation on the interfaces.

Configuring PPP/MLP MRRU Negotiation Configuration on Dialer Interfaces

In this task, you configure MRRU negotiation on the dialer interface. The bundle interface will be a virtual access interface cloned from the dialer interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface dialer** *number*
4. **ip address** *ip-address mask*
5. **encapsulation ppp**
6. **dialer** *configuration-commands*
7. **ppp multilink**
8. **ppp multilink mrru** [**local** | **remote**] *mrru-value*
9. **mtu** *bytes*
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface dialer** *number* **Example:** `Device(config)# interface dialer 1`	Defines a dialer rotary group and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.13.1.1 255.255.255.0`	Sets the IP address for the interface.

	Command or Action	Purpose
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Sets the PPP encapsulation method.
Step 6	**dialer** *configuration-commands* **Example:** `Device(config-if)# dialer string 5550101`	Configures dialer interface characteristics. • The dialer commands you use depend upon your network configuration. Choose from dialer interface configuration commands such as **dialer remote-name**, **dialer idle-timeout**, **dialer string**, and **dialer pool**. See the "Configuration Examples for PPP/MLP MRRU Negotiation Configuration" and "Related Documents" sections for dialer interface configuration examples.
Step 7	**ppp multilink** **Example:** `Device(config-if)# ppp multilink`	Enables MLP on the interface.
Step 8	**ppp multilink mrru** [**local** | **remote**] *mrru-value* **Example:** `Device(config-if)# ppp multilink mrru local 1600`	Configures the MRRU value negotiated on a multilink bundle when MLP is used. • **local**—(Optional) Configures the local MRRU value. The default values for the local MRRU are the value of the multilink group interface MTU for multilink group members, and 1524 bytes for all other interfaces. • **remote**—(Optional) Configures the minimum value that the software will accept from the peer when it advertises its MRRU. By default, the software accepts any peer MRRU value of 128 or higher. You can specify a higher minimum acceptable MRRU value in a range from 128 to 16384 bytes.
Step 9	**mtu** *bytes* **Example:** `Device(config-if)# mtu 1600`	(Optional) Adjusts the maximum packet size or MTU size. • The default MTU for serial interfaces is 1500.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

Troubleshooting Tips

Use the **debug ppp negotiation** command to verify and troubleshoot MRRU negotiation on multilink groups. Use the **show interface** command to verify MRRU negotiation on dialer interfaces.

Configuration Examples for PPP/MLP MRRU Negotiation Configuration

This section contains the following examples:

PPP/MLP MRRU Negotiation Configuration on Virtual Templates: Example

The following example shows how to configure MRRU negotiation on a virtual template with synchronous serial interfaces. The example also applies to asynchronous serial interfaces.

```
multilink virtual-template 1
!
interface virtual-template 1
 ip address 10.13.1.1 255.255.255.0
 mtu 1600
!
interface serial 0/0
 ppp multilink
 ppp multilink mrru local 1600
 mtu 1600
!
interface serial 0/1
 ppp multilink
 ppp multilink mrru local 1600
 mtu 1600
```

PPP/MLP MRRU Negotiation Configuration on Multilink Groups: Example

The following example shows how to configure MRRU negotiation on multilink groups:

```
interface multilink 10
 ip address 10.13.1.1 255.255.255.0
 ppp multilink mrru local 1600
 mtu 1600
!
interface serial 0/0
 ppp multilink
 multilink-group 10
 mtu 1600
!
interface serial 0/1
 ppp multilink
 multilink-group 10
 mtu 1600
```

PPP/MLP MRRU Negotiation Configuration on Dialer Interfaces: Example

The following example shows how to configure MRRU negotiation on dialer interfaces:

```
interface dialer 1
 ip address 10.13.1.1 255.255.255.0
 encapsulation ppp
 dialer remote-name 2610-2
 dialer idle-timeout 30 inbound
 dialer string 5550101
 dialer pool 1
 dialer-group 1
 no cdp enable
 ppp multilink
 ppp multilink mrru local 1600
 mtu 1600
```

Additional References

The following sections provide references related to the PPP/MLP MRRU Negotiation Configuration feature.

Related Documents

Related Topic	Document Title
Configuring media-independent PPP and Multilink PPP	"Part 9: PPP Configuration" in the *Cisco IOS Dial Technologies Configuration Guide*
PPP and MLP commands	*Cisco IOS Dial Technologies Command Reference*

Standards

Standards	Title
None	—

MIBs

MIBs	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

RFCs	Title
RFC 1990	*The PPP Multilink Protocol (MP)*

Technical Assistance

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

Command Reference

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Dial Technologies Command Reference* at http://www.cisco.com/en/US/docs/ios/dial/command/reference/dia_book.html. For information about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

- **ppp multilink mrru**

Feature Information for PPP/MLP MRRU Negotiation Configuration

Table 1 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.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 1 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 1 Feature Information for PPP/MLP MRRU Negotiation Configuration

Feature Name	Releases	Feature Information
PPP/MLP MRRU Negotiation Configuration	12.3(7)T 12.0(28)S 12.2(27)SB 12.2(25)S1 12.2(28)SB 12.2(33)SRC Cisco IOS Release XE 3.14S	The PPP/MLP MRRU Negotiation Configuration feature allows a device to send and receive frames over MLP bundles that are larger than the default MRRU limit of 1524 bytes. In Cisco IOS Release XE 3.14S, support was added for the Cisco 4000 series Integrated Services Routers.

Multiclass Multilink PPP

Feature History

Release	Modification
12.2(13)T	This feature was introduced.
Cisco IOS Release XE 3.14S	In Cisco IOS Release XE 3.14S, support for this feature was added for Cisco integrated services routers.

This document describes the Multiclass Multilink PPP feature in Cisco IOS Release 12.2(13)T. It includes the following sections:

Feature Overview

Previous implementations of Cisco IOS Multilink PPP (MLP) include support for Link Fragmentation Interleaving (LFI). This feature allows the delivery of delay-sensitive packets, such as the packets of a Voice call, to be expedited by omitting the PPP Multilink Protocol header and sending the packets as raw PPP packets in between the fragments of larger data packets. This feature works well on bundles consisting of a single link. However, when the bundle contains multiple links there is no way to keep the interleaved packets in sequence with respect to each other.

The Multiclass Multilink PPP (MCMP) feature in Cisco IOS Release 12.2(13)T addresses the limitations of MLP LFI on bundles containing multiple links by introducing multiple data classes. Normal data traffic and delay-sensitive data traffic are divided into Class 0 and Class 1, respectively. Class 0 data traffic is subject to fragmentation just as regular multilink packets are. Class 1 data traffic can be interleaved but never fragmented. The next transmit sequence number, expected sequence number, unassigned fragment list, working packet, lost fragment timer, fast-switching mode, and all statistics are managed per class, rather than for the bundle as a whole.

Benefits

The Multiclass Multilink PPP feature in Cisco IOS Release 12.2(13)T allows rapid delivery of real-time data over a bundle containing multiple links without loss of sequencing.

Restrictions

The **ppp multilink multiclass** command must be configured on each link that will be joining the bundle. Failure to configure this command could result in the peer refusing to allow mismatched links to join the bundle. The first link to join the bundle will determine whether MCMP is in effect for the bundle. Each subsequent link must negotiate the same MCMP parameters in order to join the bundle.

Because real-time traffic is encapsulated with multilink headers, the receiver will be required to buffer the packets when they arrive out of sequence. Therefore, the differential delay between the links must be small relative to the tolerable delay for such traffic. Otherwise, packets may be subject to additional delay while the receiver awaits the arrival of earlier sequence numbers sent over other links in the bundle.

The maximum number of links supported for an MCMP bundle is 64.

The Prefix Elision option specified in RFC 2686 is not supported.

Related Features and Technologies

- Multilink PPP

Related Documents

- *Cisco IOS Dial Technologies Configuration Guide,* Release 12.2
- *Cisco IOS Dial Technologies Command Reference*, Release 12.2

Supported Platforms

- Cisco 2600 series
- Cisco 3600 series
- Cisco 3700 series
- Cisco 7200 series
- Cisco AS5300
- Cisco AS5350
- Cisco AS5400

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 quickly 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. 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

None

MIBs

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://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. 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

- RFC 2686, *The Multi-Class Extension to Multi-Link PPP*

Prerequisites

The dialer interface, BRI interface, PRI interface, multilink interface, or virtual template must be configured, and PPP encapsulation must be enabled. For information on completing these tasks, refer to the *Cisco IOS Dial Technologies Configuration Guide,* Release 12.2.

MLP LFI must be configured on the bundle. See the section "Configuring MLP LFI on a Bundle" in this document.

Configuration Tasks

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

- Configuring MLP LFI on a Bundle (required)
- Configuring MCMP on a Member Link (required)
- Verifying MCMP (optional)

Configuring MLP LFI on a Bundle

To configure MLP LFI on a dialer, multilink, or virtual template, use the following commands beginning in interface configuration mode:

	Command	Purpose
Step 1	`Device(config-if)# ppp multilink`	Enables MLP.
Step 2	`Device(config-if)# ppp multilink interleave`	Enables interleaving of packets among the fragments of larger packets on an MLP bundle.
Step 3	`Device(config-if)# ppp multilink fragment delay milliseconds`	Specifies a maximum size, in units of time, for packet fragments on an MLP bundle.
Step 4	`Device(config-if)# ip rtp reserve lowest-udp-port range-of-ports [maximum-bandwidth]`	Reserves a special queue for real-time packet flows to specified destination User Datagram Protocol (UDP) ports, allowing real-time traffic to have higher priority than other flows.
Step 5	`Device(config-if)# exit`	Exits interface configuration mode.
Step 6	`Device(config)# multilink virtual-template number`	For virtual templates only, applies the virtual template to the multilink bundle.[1]

1. This step is not used for dialer interfaces.

Configuring MCMP on a Member Link

To configure MCMP on a configured and operational member link, use the following commands in interface configuration mode:

	Command	Purpose
Step 7	`Device(config-if)# ppp multilink`	Enables MLP.
Step 8	`Device(config-if)# ppp multilink multiclass`	Enables MCMP on an interface.

Verifying MCMP

To verify that the Multiclass Multilink PPP feature is configured correctly, enter the **show ppp multilink** EXEC command. The following output includes class-specific information for the PPP Multilink bundles:

```
Device# show ppp multilink

Virtual-Access3, bundle name is bundle1
Bundle up for 01:59:35, 1/255 load, 2 receive classes, 2 transmit classes
Receive buffer limit 12192 bytes per class, frag timeout 1524 ms
Dialer interface is Dialer1
!
Receive Class 0:
0/0 fragments/bytes in reassembly list
0 lost fragments, 0 reordered
0/0 discarded fragments/bytes, 0 lost received
0x0 received sequence
!
Receive Class 1:
0/0 fragments/bytes in reassembly list
0 lost fragments, 0 reordered
0/0 discarded fragments/bytes, 0 lost received
0x0 received sequence
!
Transmit Class 0:
0x8 sent sequence
!
Transmit Class 1:
0x0 sent sequence
!
Member links: 1 (max not set, min not set)
BR2/0:1, since 01:59:35, 80 weight, 72 frag size
```

Configuration Examples

This section provides the following configuration example:

- Configuring MCMP on a Dialer Example
- MCMP and MLP Interleaving and Queueing for Real-Time Traffic Examples

Configuring MCMP on a Dialer Example

The following partial example configures a dialer for MCMP; it does not show the configuration of the physical interfaces:

```
interface Dialer0
 ip address 10.0.0.2 255.0.0.0
 encapsulation ppp
 dialer in-band
 dialer idle-timeout 500
 dialer map ip 10.0.0.1 name remote broadcast 81012345678901
 dialer load-threshold 30 either
 dialer-group 1
 ppp authentication chap
 ppp multilink
 ppp multilink multiclass
```

MCMP and MLP Interleaving and Queueing for Real-Time Traffic Examples

The following example enables MLP interleaving and MCMP on a dialer interface that controls a rotary group of BRI interfaces. This configuration permits IP packets to trigger calls.

```
interface BRI 0
 description connected into a rotary group
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 1
 no ip address
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 2
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 3
 no ip address
 encapsulation ppp
 dialer rotary-group 1
!
interface BRI 4
 encapsulation ppp
 dialer rotary-group 1
!
interface Dialer 0
 description Dialer group controlling the BRIs
 ip address 10.1.1.1 255.255.255.0
 encapsulation ppp
 dialer map ip 10.1.1.2 name remote 14802616900
 dialer-group 1
 ppp authentication chap
! Enables Multilink Multiclass PPP interleaving on the dialer interface and reserves
! a special queue.
 ppp multilink
 ppp multilink multiclass
 ppp multilink interleave
 ip rtp reserve 32768 20 1000
! Keeps fragments of large packets small enough to ensure delay of 20 ms or less.
 ppp multilink fragment delay 20
dialer-list 1 protocol ip permit
```

The following example defines a virtual interface template that enables MLP interleaving and a maximum real-time traffic delay of 20 milliseconds. The bundle interface will be a virtual access interface cloned from the virtual template. MCMP is then configured on a member link, Serial0.

```
interface virtual-template 1
 ip unnumbered ethernet 0
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment delay 20
 ip rtp interleave 32768 20 1000
!
multilink virtual-template 1
!
interface Serial0
 encapsulation ppp
 ppp authentication chap
 ppp multilink
 ppp multilink multiclass
```

The following example configures MLP interleaving and a maximum real-time traffic delay of 20 milliseconds on a multilink interface. MCMP is then configured on a member link, Serial1, and the member link is restricted to joining only the designated multilink group interface.

```
interface Multilink1
 ip address 10.2.3.4 255.255.255.0
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment delay 20
!
interface Serial1
 encapsulation ppp
 ppp authentication chap
 ppp multilink
 ppp multilink multiclass
 ppp multilink group 1
```

Command Reference

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Dial Technologies Command Reference* at http://www.cisco.com/en/US/docs/ios/dial/command/reference/dia_book.html. For information about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

coincidental.

L2TP Large-Scale Dial-Out per-User Attribute via AAA

First Published: March 16, 2012
Last Updated: November 20, 2014

This feature makes it possible for IP per-user attributes to be applied to a Layer 2 Tunneling Protocol (L2TP) dial-out session.

Feature Specifications for L2TP Large-Scale Dial-Out per-User Attribute via AAA

Feature History	
Release	**Modification**
12.2(15)T	This feature was introduced.
Cisco IOS Release XE 3.9S	In Cisco IOS XE Release 3.9S, support was added for the Cisco CSR 1000V.
Supported Platforms	
Cisco 7200, Cisco 7400	

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

Restrictions for Using L2TP Large-Scale Dial-Out per-User Attribute via AAA

The L2TP Large-Scale Dial-Out per-User Attribute via AAA feature does *not* support the following features associated with L2TP dial-out:

- Dialer Watch

- Dialer backup
- Dialer redial
- Dialer multiple number dial
- Callback initiated by an L2TP network server (LNS), the Bandwidth Allocation Protocol (BAP), and so on

Information About L2TP Large-Scale Dial-Out per-User Attribute via AAA

To configure the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature, you need to understand the following concept:

- How the L2TP Large-Scale Dial-Out per-User Attribute via AAA Feature Works, page 1172

How the L2TP Large-Scale Dial-Out per-User Attribute via AAA Feature Works

The L2TP Large-Scale Dial-Out per-User Attribute via AAA feature makes it possible for IP and other per-user attributes to be applied to an L2TP dial-out session from an LNS. Before this feature was released, IP per-user configurations from authentication, authorization, and accounting (AAA) servers were not supported; the IP configuration would come from the dialer interface defined on the device.

The L2TP Large-Scale Dial-Out per-User Attribute via AAA feature works in a way similar to virtual profiles and L2TP dial-in. The L2TP virtual access interface is first cloned from the virtual template, which means that configurations from the virtual template interface will be applied to the L2TP virtual access interface. After authentication, the AAA per-user configuration is applied to the virtual access interface. Because AAA per-user attributes are applied only after the user has been authenticated, the LNS must be configured to authenticate the dial-out user (configuration authentication is needed for this feature).

With the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature, all software components can now use the configuration present on the virtual access interface rather than what is present on the dialer interface. For example, IP Control Protocol (IPCP) address negotiation uses the local address of the virtual access interface as the device address while negotiating with the peer.

All Cisco IOS commands that can be configured as AAA per-user commands are supported by the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature. Following is a list of some of the commands that are typically configured on a per-user basis:

- The **ip vrf forwarding** interface configuration command
- The **ip unnumbered loopback0** interface configuration command
- Per-user static routes
- Access lists
- Multilink bundles
- Idle timers

How to Configure L2TP Large-Scale Dial-Out per-User Attribute via AAA

This section contains the following procedures:

- Configuring the VPDN Group on the LNS, page 1173 (required)
- Verifying the Configuration on the Virtual Access Interface, page 1175 (optional)
- Troubleshooting the Configuration on the Virtual Access Interface, page 1175 (optional)

Configuring the VPDN Group on the LNS

You will need to configure the virtual template under the request dial-out configuration. You will also need to select the tunneling protocol and assign the virtual private dial-up network (VPDN) subgroup to a rotary group.

AAA per-user configuration is supported only on legacy dialer or dialer rotary groups and does not make sense on dialer profiles.

Be sure to configure the virtual template so that the LNS authenticates the dial-out user.

If a virtual template is not configured, L2TP dial-out per-user is not supported, but the configuration is backward compatible for all IP configurations that come from the dialer interface.

Prerequisites

The L2TP Large-Scale Dial-Out per-User Attribute via AAA feature provides additional functionality for large-scale dial-out networks and Layer 2 tunneling. It is assumed that a network is already configured and operational, and that the tasks in this document will be performed on an operational network. See the "Additional References" section for more information about large-scale dial-out networks, Layer 2 tunneling, and virtual template interfaces.

Restrictions

If the tasks in this section are not performed, the software will operate in the original mode, that is, IP per-user configurations from a AAA server will not be recognized and IP addresses will come from the dialer interface defined on the device.

To configure the VPDN group that makes it possible for IP per-user attributes to be applied to an L2TP dial-out session, use the following commands:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vpdn-group** *name*
4. **request-dialout**
5. **protocol l2tp**
6. **rotary-group** *group-number*
7. **virtual-template** *template-number*

8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vpdn-group** *name* **Example:** `Device(config)# vpdn-group 1`	Creates a VPDN group and starts VPDN group configuration mode.
Step 4	**request-dialout** **Example:** `Device(config-vpdn)# request-dialout`	Enables an LNS to request VPDN dial-out calls by using L2TP, and starts VPDN request-dialout configuration mode.
Step 5	**protocol l2tp** **Example:** `Device(config-vpdn-req-ou)# protocol l2tp`	Specifies the L2TP tunneling protocol.
Step 6	**rotary-group** *group-number* **Example:** `Device(config-vpdn-req-ou)# rotary-group 1`	Assigns a request-dialout VPDN subgroup to a dialer rotary group.
Step 7	**virtual-template** *template-number* **Example:** `Device(config-vpdn-req-ou)# virtual-template 1`	Clones the configuration from a corresponding virtual template interface, and supports IP per-user configurations from a AAA server.
Step 8	**exit** **Example:** `Device(config-vpdn-req-ou)# exit`	Exits VPDN request-dialout configuration mode.

What to Do Next

The configuration for the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature must include a AAA profile to specify the per-user attributes. See the "Per-User AAA Attributes Profile Example" for an example of such a profile.

Verifying the Configuration on the Virtual Access Interface

This task verifies that the per-user AAA commands are successfully parsed on the virtual access interface.

SUMMARY STEPS

1. **enable**
2. **show interfaces virtual-access** *number* [**configuration**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show interfaces virtual-access** *number* [**configuration**] **Example:** `Device# show interfaces virtual-access 3 configuration`	Displays status, traffic data, and configuration information about a specified virtual access interface. • **configuration**—(Optional) Restricts output to configuration information.

Troubleshooting the Configuration on the Virtual Access Interface

This task displays additional information about the per-user AAA commands that are parsed on the virtual access interface.

SUMMARY STEPS

1. Attach a console directly to a device.
2. **enable**
3. **configure terminal**
4. **no logging console**
5. Use Telnet to access a device port and repeat Steps 2 and 3.
6. **terminal monitor**
7. **exit**
8. **debug aaa per-user**
9. **debug vtemplate events**
10. **debug vtemplate cloning**
11. **configure terminal**
12. **no terminal monitor**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Attach a console directly to a device.	—
Step 2	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**no logging console** **Example:** `Device(config)# no logging console`	Disables all logging to the console terminal. • To reenable logging to the console, use the **logging console** command in global configuration mode.
Step 5	Use Telnet to access a Device port and repeat Steps 2 and 3.	Enters global configuration mode in a recursive Telnet session, which allows the output to be redirected away from the console port.
Step 6	**terminal monitor** **Example:** `Device(config)# terminal monitor`	Enables logging output on the virtual terminal.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.
Step 8	**debug aaa per-user** **Example:** `Device# debug aaa per-user`	Displays what attributes are applied to each user as the user authenticates.
Step 9	**debug vtemplate events** **Example:** `Device# debug vtemplate events`	Displays the virtual template events to form a virtual access interface.
Step 10	**debug vtemplate cloning** **Example:** `Device# debug vtemplate cloning`	Displays the virtual template cloning to form a virtual access interface. • Use this command to verify when the interface is created (cloned from the virtual template) at the beginning of the dialup connection and when the interface is destroyed when the connection is terminated.

	Command or Action	Purpose
Step 11	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 12	**no terminal monitor** **Example:** `Device(config)# no terminal monitor`	Disables logging on the virtual terminal.
Step 13	**exit** **Example:** `DeviceDevice(config)# exit`	Exits to privileged EXEC mode.

Configuration Examples for L2TP Large-Scale Dial-Out per-User Attribute via AAA

This section provides the following configuration examples to show how to configure the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature:

LNS Configuration Example

The following partial example shows how to configure an LNS for the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature:

```
!
vpdn enable
vpdn search-order domain
!
vpdn-group 1
.
.
.
 request-dialout
  protocol l2tp
  rotary-group 1
  virtual-template 1
 initiate-to ip 10.0.1.194.2
 local name lns
 l2tp tunnel password 7094F3$!5^3
 source-ip 10.0.194.53
!
```

Per-User AAA Attributes Profile Example

The following example shows the attribute-value pair (avpair) statements for a AAA profile to specify the per-user attributes:

```
5300-Router1-out  Password = "cisco"
     Service-Type = Outbound
     cisco-avpair = "outbound:dial-number=5553021"
7200-Router1-1  Password = "cisco"
     Service-Type = Outbound
     cisco-avpair = "ip:route=10.17.17.1 255.255.255.255 Dialer1 100 name 5300-Router1"
5300-Router1 Password = "cisco"
     Service-Type = Framed
     Framed-Protocol = PPP
     cisco-avpair = "lcp:interface-config=ip unnumbered loopback 0"
     cisco-avpair = "ip:outacl#1=deny ip host 10.5.5.5 any log"
     cisco-avpair = "ip:outacl#2=permit ip any any"
     cisco-avpair = "ip:inacl#1=deny ip host 10.5.5.5 any log"
     cisco-avpair = "ip:inacl#2=permit ip any any"
     cisco-avpair = "multilink:min-links=2"
     Framed-Route = "10.5.5.6/32 Ethernet4/0"
     Framed-Route = "10.5.5.5/32 Ethernet4/0"
     Idle-Timeout = 100
```

Virtual Access Interface Configuration Verification Example

The following example shows the virtual access interface configuration so you can check that the per-user AAA commands are correctly parsed:

```
Device# show interfaces virtual-access 3 configuration

Virtual-Access3 is an VPDN link (sub)interface

Derived configuration : 212 bytes
!
interface Virtual-Access3
 ip vrf forwarding V1.25.com
 ip unnumbered Loopback25
 no peer default ip address
 ppp authentication chap
end
```

Virtual Access Interface Configuration Troubleshooting Example

This section provides the following debugging session examples for a network configured with the L2TP Large-Scale Dial-Out per-User Attribute via AAA feature. Output is displayed for each command in the task.

Sample Output for the debug aaa per-user Command

```
Device# debug aaa per-user

%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up
AAA/AUTHOR: Processing PerUser AV interface-config
AAA/AUTHOR: Processing PerUser AV route
AAA/AUTHOR: Processing PerUser AV route
AAA/AUTHOR: Processing PerUser AV outacl
AAA/AUTHOR: Processing PerUser AV outacl
```

```
AAA/AUTHOR: Processing PerUser AV inacl
AAA/AUTHOR: Processing PerUser AV inacl
Vi3 AAA/PERUSER/ROUTE: vrf name for vaccess: V1.25.com
Vi3 AAA/PERUSER/ROUTE: route string: IP route vrf V1.25.com 10.1.25.10 255.255.255.255 10.1.25.20 tag 120
Vi3 AAA/PERUSER/ROUTE: vrf name for vaccess: V1.25.com
Vi3 AAA/PERUSER/ROUTE: route string: IP route vrf V1.25.com 172.30.35.0 255.255.255.0 10.1.25.20 tag 120

AAA/PER-USER: mode = config; command = [ip access-list extended Virtual-Access3#41
permit icmp any any log
permit ip any any]
AAA/PER-USER: line = [ip access-list extended Virtual-Access3#41]
AAA/PER-USER: line = [permit icmp any any log]
AAA/PER-USER: line = [permit ip any any]
AAA/PER-USER: mode = config; command = [ip access-list extended Virtual-Access3#42
permit icmp any any log
permit ip any any]
AAA/PER-USER: line = [ip access-list extended Virtual-Access3#42]
AAA/PER-USER: line = [permit icmp any any log]
AAA/PER-USER: line = [permit ip any any]
AAA/PER-USER: mode = config; command = [IP route vrf V1.25.com 10.1.25.10 255.255.255.255 10.1.25.20 tag
120 IP route vrf V1.25.com 172.30.35.0 255.255.255.0 10.1.25.20 tag 120]
AAA/PER-USER: line = [IP route vrf V1.25.com 10.1.25.10 255.255.255.255 10.1.25.20 tag 120]
AAA/PER-USER: line = [IP route vrf V1.25.com 172.30.35.0 255.255.255.0 10.1.25.20 tag 120]
*Feb 28 07:35:19.616: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to
up
```

Sample Output for the debug vtemplate events and debug vtemplate cloning Commands

```
Device# debug vtemplate events
Device# debug vtemplate cloning

VT[Vi3]:Reuse interface, recycle queue size 1
VT[Vi3]:Set to default using 'encap ppp'
VT[Vi3]:Vaccess created
VT[Vi3]:Added new vtemplate cloneblk, now cloning from vtemplate
VT[Vi3]:Clone Vaccess from Virtual-Template25 (19 bytes)
VT[Vi3]:no ip address
VT[Vi3]:end
VT[Vi3]:Applying config commands on process "Dialer event" (25)
VT[Vi3]:no ip address
VT[Vi3]:end
%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up
VT:Sending vaccess request, id 0x6401947C
VT:Processing vaccess requests, 1 outstanding
VT[Vi3]:Added new AAA cloneblk, now cloning from vtemplate/AAA
VT[Vi3]:Clone Vaccess from AAA (60 bytes)
VT[Vi3]:ip vrf forwarding V1.25.com
VT[Vi3]:ip unnumbered loopback25
VT[Vi3]:end
VT[Vi3]:Applying config commands on process "VTEMPLATE Background Mgr" (160)
VT[Vi3]:ip vrf forwarding V1.25.com
VT[Vi3]:ip unnumbered loopback25
VT[Vi3]:end
VT[Vi3]:MTUs ip 1500, sub 0, max 1500, default 1500
VT[Vi3]:Processing vaccess response, id 0x6401947C, result success (1)
VT[Vi3]:Added new AAA cloneblk, now cloning from vtemplate/AAA
VT[Vi3]:Clone Vaccess from AAA (82 bytes)
VT[Vi3]:IP access-group Virtual-Access3#51 in
VT[Vi3]:IP access-group Virtual-Access3#52 out
VT[Vi3]:end
VT[Vi3]:Applying config commands on process "PPP IP Route" (62)
VT[Vi3]:IP access-group Virtual-Access3#51 in
VT[Vi3]:IP access-group Virtual-Access3#52 out
VT[Vi3]:end
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to up
```

Additional References

For additional information related to L2TP large-scale dial-out per-user attributes using a AAA server, see to the following sections:

Related Documents

Related Topic	Document Title
Large-scale dial-out	*Cisco IOS Dial Technologies Configuration Guide*, Release 12.2*; refer to the chapter* "Configuring Large-Scale Dial-Out."
VPDN groups	*Cisco IOS Dial Technologies Configuration Guide*, Release 12.2*; refer to the chapter* "Configuring Virtual Private Networks."
Virtual interfaces	*Cisco IOS Dial Technologies Configuration Guide*, Release 12.2*; refer to the chapter* "Configuring Virtual Template Interfaces."
Per-user configuration	*Cisco IOS Dial Technologies Configuration Guide*, Release 12.2*; refer to the chapter* "Configuring Per-User Configuration."
Descriptions of debug command output	*Cisco IOS Debug Command Reference*, Release 12.2.

Standards

Standards	Title
None	—

MIBs

MIBs	MIBs Link
None	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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. 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

RFCs	Title
None	—

Technical Assistance

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

Command Reference

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Dial Technologies Command Reference* at http://www.cisco.com/en/US/docs/ios/dial/command/reference/dia_book.html. For information about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

- **virtual-template**

```
x25 route 11111 interface Dialer0
x25 route 44444 interface Dialer0
!
```

Virtual Interface Template Service

First Published: May 10, 2001
Last Updated: November 20, 2014

The Virtual Interface Template Service feature provides a generic service that can be used to apply predefined interface configurations (virtual interface template services) in creating and freeing virtual access interfaces dynamically, as needed.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Virtual Interface Template" section on page 916.

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

Contents

Restrictions for Virtual Interface Template Service

The following restrictions apply for configuring the virtual interface template service feature:

- Although a system can generally support many virtual interface template services, one template for each virtual access application is a more realistic limit.
- When in use, each virtual access interface cloned from a template requires the same amount of memory as a serial interface. Limits to the number of virtual access interfaces that can be configured are determined by the platform.
- You cannot reuse virtual interface templates. You need to create different templates for different interface configurations.
- You cannot directly configure virtual access interfaces. You need to configure a virtual access interface by configuring a virtual interface template service or including the configuration information of the user on an authentication, authorization, and accounting (AAA) server. However, information about an in-use virtual access interface can be displayed, and the virtual access interface can be cleared.
- Virtual interface templates provide no *direct* value to you; they must be applied to or associated with a virtual access feature using a command with the **virtual-template** keyword.

  For example, the **interface virtual-template** command creates the virtual interface template service.

For a complete description of the virtual interface service commands mentioned in this chapter, refer to the *Cisco IOS Dial Technologies Command Reference*. For information about all Cisco IOS commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or a Cisco IOS master commands list.

Information About Virtual Interface Template Service

To configure the virtual interface template service, you should understand the following concepts:

- Virtual Interface Template Service Overview, page 908
- Benefits of Virtual Interface Template Service, page 909
- Features that Use Virtual Interface Template Service, page 909
- Selective Virtual Access Interface Creation, page 910

Virtual Interface Template Service Overview

Virtual interface template services can be configured independently of any physical interface and applied dynamically, as needed, to create virtual access interfaces. When a user dials in, a predefined configuration template is used to configure a virtual access interface; when the user is done, the virtual access interface goes down and the resources are freed for other dial-in uses.

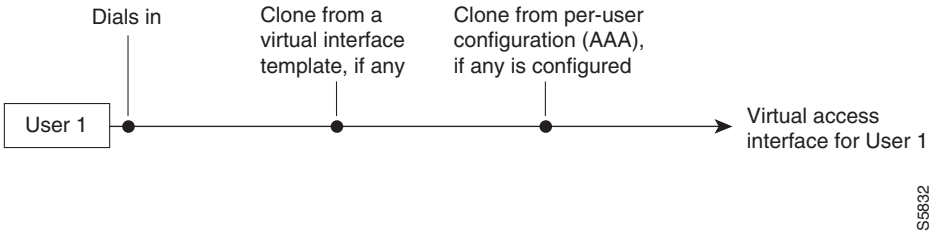
A virtual interface template service is a logical entity—a configuration for a serial interface but not tied to a physical interface—that can be applied dynamically as needed. Virtual access interfaces are virtual interfaces that are created, configured dynamically (for example, by *cloning* a virtual interface template service), used, and then freed when no longer needed.

Virtual interface template services are one possible source of configuration information for a virtual access interface.

Each virtual access interface can clone from only one template. But some applications can take configuration information from multiple sources; The result of using template and AAA configuration sources is a virtual access interface uniquely configured for a specific dial-in user.

Figure 1 illustrates that a device can create a virtual access interface by first using the information from a virtual interface template service (if any is defined for the application) and then using the information in a per-user configuration.

Figure 1 Possible Configuration Sources for Virtual Access Interfaces



Benefits of Virtual Interface Template Service

The virtual interface template service is intended primarily for customers with large numbers of dial-in users and provides the following benefits:

- Easy maintenance: It allows customized configurations to be predefined and then applied dynamically when the specific need arises.
- Scalability: It allows interface configuration to be separated from physical interfaces. Virtual interfaces can share characteristics, no matter what specific type of interface the user called on.
- Consistency and configuration ease: It allows the same predefined template to be used for all users dialing in for a specific application.
- Efficient device operation: It frees the virtual access interface memory for another dial-in use when the call from the user ends.

Features that Use Virtual Interface Template Service

The following features use virtual interface template service to create virtual access interfaces dynamically:

- Virtual Private Dialup Networks (VPDNs)
- Virtual interface templates for protocol translation
- PPP over ATM

Virtual interface templates are supported on all platforms that support these features.

To create and configure a virtual interface template interface, compete the tasks in the "Creating and Configuring a Virtual Interface Template" section on page 910. To apply a virtual interface template service, refer to the specific feature that applies the virtual interface template.

All prerequisites depend on the feature that is applying a virtual interface template to create a virtual access interface. Virtual interface template services themselves have no other prerequisites.

Selective Virtual Access Interface Creation

You can configure a device to automatically determine whether to create a virtual access interface for each inbound connection. In particular, a call that is received on a physical asynchronous interface that uses a AAA per-user configuration for RADIUS or TACACS+ can be processed without a virtual access interface being created by a device.

To determine whether a virtual access interface is created, ensure the following exists:

- AAA per-user configuration
- Support for link interface support direct per-user AAA

A virtual access interface is created if there is a AAA per-user configuration *and* the link interface does not support direct per-user AAA (such as ISDN).

A virtual access interface is not created if the following conditions are not satisfied:

- There is no AAA per-user configuration.
- There is AAA per-user configuration and the link interface does support direct per-user AAA (such as asynchronous).

How to Configure a Virtual Interface Template

This section contains the following tasks:

- Creating and Configuring a Virtual Interface Template, page 910 (required)
- Monitoring and Maintaining a Virtual Access Interface, page 911 (required)

Note The order in which you create virtual interface template service and configure the features that use the templates and profiles is not important. They must exist, however, before someone calling in can use them.

Creating and Configuring a Virtual Interface Template

To create and configure a virtual interface template service, use the **interface virtual-template** command.

Note Configuring the **ip address** command within a virtual interface template service is not recommended. Configuring a specific IP address in a virtual interface template can result in the establishment of erroneous routes and the loss of IP packets.

Other PPP configuration commands can be added to the virtual interface template configuration. For example, you can add the **ppp authentication chap** command.

All configuration commands that apply to serial interfaces can also be applied to virtual interface template interfaces, except the **shutdown** and **dialer** commands.

For virtual interface template examples, see the "Configuration Examples for Virtual Interface Template" section on page 912 section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual-template** *number*
4. **ip unnumbered ethernet** *number*
5. **encapsulation ppp**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual-template** *number* **Example:** `Device(config)# interface virtual-template 0/0`	Creates a virtual interface template and enters interface configuration mode.
Step 4	**ip unnumbered ethernet** number **Example:** `Device(config-if)# ip unnumbered ethernet 0/0`	Enables IP without assigning a specific IP address on the LAN.
Step 5	**encapsulation ppp** **Example:** `Device(config-if)# encapsulation ppp`	Enables PPP encapsulation on the virtual interface template.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring and Maintaining a Virtual Access Interface

When a virtual interface template or a configuration from a user on a AAA server or both are applied dynamically, a virtual access interface is created. Although a virtual access interface cannot be created and configured directly, it can be displayed and cleared.

To display or clear a specific virtual access interface, use the **show interfaces virtual-access** and c**lear interface virtual-access** commands.

SUMMARY STEPS

1. **enable**
2. **show interfaces virtual-access** *number*
3. **clear interface virtual-access** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show interfaces virtual-access** *number* **Example:** `Device# show interfaces virtual-access 3`	Displays the configuration of the virtual access interface.
Step 3	**clear interface virtual-access** *number* **Example:** `Device# clear interface virtual-access 3`	Tears down the virtual access interface and frees the memory for other dial-in uses.

Configuration Examples for Virtual Interface Template

The following sections provide virtual interface template configuration examples:

- Virtual Interface Template: Example, page 912
- Selective Virtual Access Interface: Example, page 913
- Selective Virtual Access Interface Configuration for RADIUS per User: Example, page 913
- Selective Virtual Access Interface Configuration for TACACS+ per User: Example, page 913

Virtual Interface Template: Example

The following example shows how to verify a virtual interface template configuration.

Note Effective with Cisco Release 12.4(11)T, the **l2f protocol** command was removed in Cisco IOS software.

```
Device# show interfaces virtual-access 1

Virtual-Access1 is a L2F link interface
interface Virtual-Access1 configuration...
ip unnumbered ethernet0
ipx ppp-client Loopback2
no cdp enable
ppp authentication chap
```

Selective Virtual Access Interface: Example

The following example shows how to create a virtual access interface for incoming calls that require a virtual access interface:

```
aaa new-model
aaa authentication ppp default local radius tacacs
aaa authorization network default local radius tacacs

virtual-profile if-needed
virtual-profile virtual-template 1
virtual-profile aaa
!
interface virtual-template 1
 ip unnumbered Ethernet 0
 no ip directed-broadcast
 no keepalive
 ppp authentication chap
 ppp multilink
```

Selective Virtual Access Interface Configuration for RADIUS per User: Example

This example shows how to create AAA per-user configuration for a RADIUS user profile. When a AAA per-user configuration for a RADIUS user profile exists, a virtual access interface is configured automatically.

```
RADIUS user profile:
        name1 Password = "test"
                 User-Service-Type = Framed-User,
                 Framed-Protocol = PPP,
                cisco-avpair = "ip:inacl#1=deny 10.10.10.10 0.0.0.0",
                cisco-avpair = "ip:inacl#1=permit any"
```

Selective Virtual Access Interface Configuration for TACACS+ per User: Example

This example shows how to create AAA per-user configuration for a TACACS+ user profile:

```
user = name1 {
                name = "name1"
                global = cleartext test
                service = PPP protocol= ip {
                        inacl#1="deny 10.10.10.10 0.0.0.0"
                        inacl#1="permit any"
                }
        }
```

Additional References

The following sections provide references related to the Virtual Interface Template Service feature.

Related Documents

Related Topic	Document Title
Dial interfaces, controllers and lines	"Overview of Dial Interfaces, Controllers, and Lines" module in the *Cisco IOS Dial Technologies Configuration Guide*
Dial commands	*Cisco IOS Dial Technologies Command Reference*

Standards

Standard	Title
None	—

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
None	—

Technical Assistance

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

Feature Information for Virtual Interface Template

Table 1 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.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 1 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 1 Feature Information for Virtual Interface Templates

Feature Name	Releases	Feature Information
Virtual Interface Template Service	11.2(1) 12.2(14)S 12.2(27)SBA 12.2(33)SRE Cisco IOS XE 3S	Virtual interface template service can be configured independently of any physical interface and applied dynamically to create virtual access interfaces. The following sections provide information about this feature: • Information About Virtual Interface Template Service, page 908 • Creating and Configuring a Virtual Interface Template, page 910 • Monitoring and Maintaining a Virtual Access Interface, page 911 The following commands were introduced or modified: **clear interfaces virtual-access**, **interface virtual-template**, and **show interfaces virtual-access**