
Configuring SLIP and PPP

This chapter describes how to configure asynchronous interfaces to support remote nodes using Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP) encapsulation. See the *Access Services Command Reference* for a complete description of the commands listed in this chapter.

Refer to the chapter “Making Connections to Network Devices” in this book for information about EXEC user commands and establishing SLIP and PPP connections.

Cisco’s Implementation of SLIP and PPP

Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP) define methods of sending Internet Protocol (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 PCs to a network. SLIP and PPP over asynchronous dial-up modems allow a home computer to be connected to a network without the cost of a leased line. Dial-up SLIP and PPP links can also be used for remote sites that need only occasional remote node or backup connectivity. Both public-domain and vendor-supported SLIP and PPP implementations are available for a variety of computer applications.

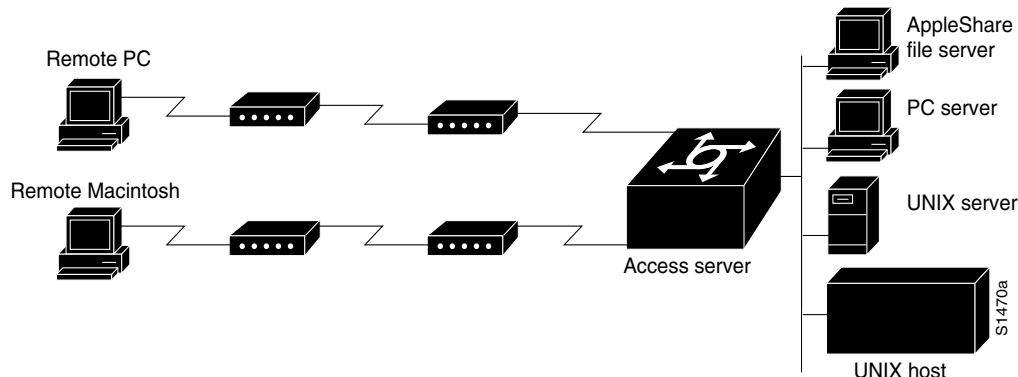
The Cisco IOS software concentrates a large number of SLIP or PPP PC or workstation client hosts onto a network interface that allows the PCs to communicate with any host on the network. The Cisco IOS software can support any combination of SLIP or PPP lines and lines dedicated to normal asynchronous devices such as terminals and modems. Refer to RFC 1055 for more information about SLIP, and RFCs 1331 and 1332 for more information about PPP.

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

Figure 28 Sample SLIP or PPP Remote-Node Configuration



Responding to BOOTP Requests

The 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 into 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, typically using TFTP.

SLIP and PPP 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 find out what its IP address is 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 SLIP and PPP 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 chapters "Configuring IP" and "Configuring IP Routing Protocols," in *Network Protocols Configuration Guide, Part 1*, for more information about RARP.) However, RARP is a hardware link protocol, so it 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.

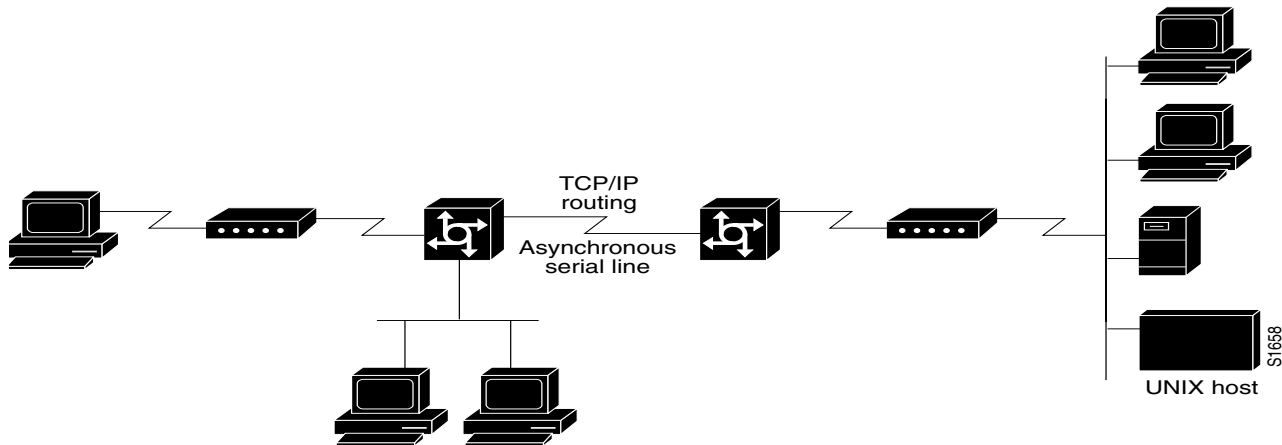
The Cisco IOS software also supports IP routing connections for communication that requires connecting one network to another.

The Cisco IOS software supports protocol translation for SLIP and PPP between other network devices running Telnet, LAT, or X.25. For example, you can send IP packets across a public X.25 PAD network using SLIP or PPP encapsulation when SLIP or PPP protocol translation is enabled. For more information, refer to the chapter "Configuring Protocol Translation" in this publication.

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

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 29 shows a sample asynchronous routing configuration.

Figure 29 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.

Remote Node Configuration Task List

To configure the Cisco IOS software through a remote node, you must perform the first task in the following list on your asynchronous interfaces. Perform the rest of the tasks to customize the asynchronous interface for your particular network environment and to monitor asynchronous connections:

- Configure Asynchronous Interfaces
- Configure Network-Layer Protocols over SLIP and PPP
- Enable SLIP and PPP on Virtual Asynchronous Interfaces
- Configure Remote Access to NetBEUI Services
- Configure Performance Parameters
- Optimize Available Bandwidth
- Specify the MTU Size of IP Packets
- Provide Backward Compatibility for SLIP and PPP
- Modify the IP Output Queue Size

- Specify IP Access Lists
- Configure Support for Extended BOOTP Requests
- Monitor and Maintain Asynchronous Interfaces
- Asynchronous Interface Configuration Examples

If you want to call back a PPP client requesting asynchronous callback, refer to the section “Call Back PPP Clients” in the “Configuring Terminal Lines and Modem Support” chapter.

The steps to perform these tasks are described in the following sections. See the “Asynchronous Interface Configuration Examples” section at the end of this chapter for examples of asynchronous configuration files. Tasks are performed in global configuration mode unless otherwise specified.

Configure Asynchronous Interfaces

To configure the Cisco IOS software through a remote node, configure basic functionality on your asynchronous interfaces, and then customize the interfaces for your environment. Basic configuration tasks include the following:

- Specify an Asynchronous Interface
- Configure SLIP or PPP Encapsulation
- Specify Dedicated or Interactive Mode
- Configure the Interface Addressing Method for Remote Devices
- Assign IP Addresses for Local Devices
- Enable Asynchronous Routing
- Enable Default Routing on an Asynchronous Interface
- Configure Automatic Protocol Startup
- Make SLIP and PPP connections at the EXEC level if you have configured interactive mode. Refer to the chapter “Making Connections to Network Devices” in this book for more information about making SLIP and PPP connections.

Specify an Asynchronous Interface

To specify an asynchronous interface, perform the following task in global configuration mode:

Task	Command
Specify an asynchronous interface.	interface async <i>unit</i>

Configure SLIP or PPP Encapsulation

SLIP and PPP are methods of encapsulating datagrams and other network-layer protocol information over point-to-point links. To configure the default encapsulation on an asynchronous interface, perform the following task in interface configuration mode:

Task	Command
Configure PPP or SLIP encapsulation on an asynchronous line.	encapsulation { ppp slip }

To support SLIP or PPP for IP, the Cisco IOS software must be configured with an IP routing protocol or with the **ip host-routing** command.

When an asynchronous interface is encapsulated with PPP or SLIP, IP fast switching is enabled. For more information on IP fast switching, refer to the “Enable Fast Switching” section later in this chapter.

Specify Dedicated or Interactive Mode

You can configure one or more asynchronous interfaces on your access server (and one on a router) to be in dedicated network interface mode. In dedicated mode, an interface is automatically configured for SLIP or PPP connections. There is no user prompt or EXEC level, and no end-user commands are required to initiate remote-node connections. If you want a line to be used only for SLIP or PPP connections, configure the line for dedicated mode.

In interactive mode, a line can be used to make any type of connection, depending on the EXEC command entered by the user. For example, depending on its configuration, the line could be used for Telnet or XRemote connections, or SLIP or PPP encapsulation. The user is prompted for an EXEC command before a connection is initiated.

This section describes the following tasks:

- Configure dedicated network mode
- Return a line to interactive mode

Configure Dedicated Network Mode

You can configure an asynchronous interface to be in dedicated network mode. When the interface is configured for dedicated mode, the end user cannot change the encapsulation method, address, or other parameters.

To configure an interface for dedicated network mode, perform the following task in interface configuration mode.

Task	Command
Place the line into dedicated asynchronous network mode.	async mode dedicated

Return a Line to Interactive Mode

After a line has been placed in dedicated mode, perform the following task in interface configuration mode to return it to interactive mode.

Task	Command
Return the line to interactive mode.	async mode interactive

By default, no asynchronous mode is configured. In this state, the line is not available for inbound networking because the SLIP and PPP connections are disabled.

Configure the Interface Addressing Method for Remote Devices

You can control whether addressing is dynamic (the user specifies the address at the EXEC level when making the connection), or whether default addressing is used (the address is forced by the system). If you specify dynamic addressing, the router must be in interactive mode and the user will enter the address at the EXEC level.

It is common to configure an asynchronous interface to have a default address and to allow dynamic addressing. With this configuration, the choice between the default address or a dynamic addressing is made by the users when they enter the **slip** or **ppp** EXEC command. If the user enters an address, it is used, and if the user enters the **default** keyword, the default address is used.

This section describes the following tasks:

- Assign a default asynchronous address.
- Allow an asynchronous address to be assigned dynamically.

Assign a Default Asynchronous Address

Perform the following task in interface configuration mode to assign a permanent default asynchronous address:

Task	Command
Assign a default IP address to an asynchronous interface.	peer default ip address <i>address</i>

Use the **no** form of this command to disable the default address. If the server has been configured to authenticate asynchronous connections, you are prompted for a password after entering the **slip default** or **ppp default** EXEC command before the line is placed into asynchronous mode.

The assigned default address is implemented when the user enters the **slip default** or **ppp default** EXEC command. The transaction is validated by the Terminal Access Controller Access System (TACACS) server, when enabled, and the line is put into network mode using the address that is in the configuration file.

Configuring a default address is useful when the user is not required to know the IP address to gain access to a system (for example, users of a server that is available to many students on a campus). Instead of requiring each user to know an IP address, they only need to enter the **slip default** or **ppp default** EXEC command and let the server select the address to use. See the chapter “Making Connections to Network Devices” in this book for more information about the **slip** and **ppp** EXEC commands.

Allow an Asynchronous Address to Be Assigned Dynamically

When a line is configured for dynamic assignment of asynchronous addresses, the user enters the **slip** or **ppp** EXEC command and is prompted for an address or logical host name. The address is validated by TACACS, when enabled, and the line is assigned the given address and put into asynchronous mode. Assigning asynchronous addresses dynamically is also useful when you want to assign set addresses to users. For example, an application on a personal computer that automatically dials in using SLIP and polls for electronic mail messages can be set up to dial in periodically and enter the required IP address and password.

To assign asynchronous addresses dynamically, perform the following task in interface configuration mode:

Task	Command
Allow the IP address to be assigned when the protocol is initiated.	async dynamic address

The dynamic addressing features of the internetwork allow packets to get to their destination and back regardless of the access server, router, or network they are sent from. For example, if a host such as a laptop computer moves from place to place it can keep the same address no matter where it is dialing in from.

Logical host names are first converted to uppercase and then sent to the TACACS server for authentication.

Assign IP Addresses for Local Devices

The local address is set using the **ip address** or **ip unnumbered** command.

IP addresses identify locations to which IP datagrams can be sent. You must assign each router interface an IP address. See the *Internetworking Technology Overview* publication for detailed information on IP addresses.

To assign an IP address to a network interface, perform the following task in interface configuration mode:

Task	Command
Set an IP address for an interface.	ip address address mask [secondary]

A subnet mask identifies the subnet field of a network address. Subnets are described in the *Internetworking Technology Overview* publication.

Conserve Network Addresses

When asynchronous routing is enabled, you might find it necessary to conserve network addresses by configuring the asynchronous interfaces as *unnumbered*. An unnumbered interface does not have an address. Network resources are therefore conserved because fewer network numbers are used and routing tables are smaller.

To configure an unnumbered interface, perform the following task in interface configuration mode:

Task	Command
Configure the asynchronous interface to be unnumbered.	ip unnumbered type number

Whenever the unnumbered interface generates a packet (for example, a routing update), it uses the address of the specified interface as the source address of the IP packet. It also uses the address of the specified interface to determine which routing processes are sending updates over the unnumbered interface.

You can use the IP unnumbered feature even if the system on the other end of the asynchronous link does not support it. The IP unnumbered feature is transparent to the other end of the link because each system bases its routing activities on information in the routing updates it receives and on its own interface address.

Enable Asynchronous Routing

To route IP packets on an asynchronous interface, perform the following task in interface configuration mode:

Task	Command
Configure an asynchronous interface for routing.	async dynamic routing

This command permits you to enable the IGRP, RIP, and OSPF routing protocols for use when the user makes a connection using the **ppp** or **slip** EXEC commands. The user must, however, specify **/routing** keyword at the SLIP or PPP command line.

Enable Default Routing on an Asynchronous Interface

To automatically enable the use of IP routing protocols on an asynchronous interfaces with the **ppp** and **slip** EXEC commands, perform the following task in interface configuration mode.

Task	Command
Automatically configure an asynchronous interface for routing.	async default routing

For asynchronous interfaces in interactive mode, the **async default routing** command causes the **ppp** and **slip** EXEC commands to be interpreted as though the **/route** switch had been included in the command. For asynchronous interfaces in dedicated mode, the **async dynamic routing** command enables routing protocols to be used on the line. Without the **async default routing** command, there is no way to enable the use of routing protocols automatically on a dedicated asynchronous interface.

Configure Automatic Protocol Startup

To configure the Cisco IOS software to allow a PPP or SLIP session to start automatically, perform the following tasks in line configuration mode:

Task	Command
Configure a line to start an ARA, PPP or SLIP session automatically.	autoselect {arap ppp slip} during login¹

1. This command is documented in the “Terminal Line and Modem Support Commands” chapter of the *Access Services Command Reference*.

The **autoselect** command permits the Cisco IOS software to allow an appropriate process to start automatically when a starting character is received. The software detects either a Return character, which is the start character for an EXEC session or the start character for the ARA protocol. By using the optional **during login** argument, the username or password prompt is displayed without pressing the Return key. While the username or password prompts are displayed, you can choose to answer these prompts or to start sending packets from an autoselected protocol. Refer to the end of this chapter for configuration examples.

Note When using autoselect, the activation character should be set to the default Return, and exec-character-bits should be set to 7. If you change these defaults, the application cannot recognize the activation request.

Configure Network-Layer Protocols over SLIP and PPP

You can configure network-layer protocols, such as AppleTalk, IP, and IPX, over SLIP and PPP. SLIP supports only IP, but PPP supports each of these protocols. Refer to the sections that follow to configure these protocols over SLIP and PPP.

You can also configure IPX–PPP on VTYs. Refer to the section “Enable IPX–PPP on Virtual Asynchronous Interfaces.”

Configure IP–SLIP

To enable IP–SLIP on a synchronous or asynchronous interface, perform the following tasks, beginning in interface configuration mode:

Task	Command
Step 1 Configure IP routing on the interface. or Configure IP unnumbered routing on a serial interface.	ip address <i>ip-address mask</i> or ip unnumbered <i>type number</i>
Step 2 Enable SLIP encapsulation on the serial interface.	encapsulation slip
Step 3 Enable interactive mode on an asynchronous interface.	async mode interactive

Configure AppleTalk–PPP

You can configure an asynchronous interface so that users can access AppleTalk zones by dialing into 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 also referred to as ATCP.

You create a virtual network that exists only for accessing an AppleTalk internet through the server. To create a new AppleTalk zone, issue 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.

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

Task	Command
Step 1 Define encapsulation as PPP on this interface.	encapsulation ppp
Step 2 Create an internal network on the server.	appletalk virtual-net <i>network-number zone-name</i>

Task	Command
Step 3 Enable client-mode on this interface.	appletalk client-mode

Configure IP–PPP

To enable IP–PPP (IPCP) on a synchronous or asynchronous interface, perform the following tasks, beginning in interface configuration mode:

Task	Command
Step 1 Configure IP routing on the interface. or Configure IP unnumbered routing on a serial interface.	ip address <i>ip-address mask</i> or ip unnumbered <i>type number</i>
Step 2 Enable PPP encapsulation on the serial interface.	encapsulation ppp
Step 3 Enable interactive mode on an asynchronous interface.	async mode interactive

Configure IPX–PPP

You can configure IPX to run 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 VTYs configured for PPP. Refer to the section “Enable IPX–PPP on Virtual Asynchronous Interfaces” later in this chapter.

IPX–PPP—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, perform the following tasks, beginning in global configuration mode:

Task	Command
Step 1 Enable IPX routing.	ipx routing [<i>node</i>]
Step 2 Create a loopback interface, which is a virtual interface, existing only inside the router.	interface loopback <i>number</i>
Step 3 Enable IPX routing on the loopback interface.	ipx network <i>network</i> ¹
Step 4 Exit to global configuration mode.	exit
Step 5 Enter interface configuration mode for the asynchronous interface.	interface async <i>number</i>
Step 6 Configure IP unnumbered routing on the interface.	ip unnumbered <i>type number</i>

Task	Command
Step 7 Enable PPP encapsulation on the interface.	encapsulation ppp
Step 8 Enable interactive mode on an asynchronous interface.	async mode interactive
Step 9 Assign the asynchronous interface to the loopback interface configured for IPX.	ipx ppp-client Loopback number
Step 10 Turn off SAP updates to optimize bandwidth on asynchronous interfaces.	ipx sap-interval 0

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

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. Step 10 in the previous task table uses the **ipx sap-interval 0** to filter SAP updates. For more information about filtering routing updates, refer to the section “Create Filters for Updating the Routing Table” in the “Configuring Novell IPX” chapter in the *Network Protocols Configuration Guide, Part 2*.

IPX–PPP—Using Dedicated IPX Network Numbers for Each Interface

To enable IPX–PPP, perform the following tasks starting in global configuration mode. The first five tasks are required. The last task is optional:

Task	Command
Step 1 Enable IPX routing.	ipx routing [node]
Step 2 Enter interface configuration mode.	interface type number
Step 3 Enable PPP encapsulation on the interface.	encapsulation ppp
Step 4 Enable interactive mode on an asynchronous interface.	async mode interactive
Step 5 Enable IPX routing on the interface.	ipx network network¹
Step 6 Turn off SAP updates to optimize bandwidth on asynchronous interfaces.	ipx sap-interval 0

1. Every interface must have a *unique* ipx network number.

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. To filter routing updates, refer to the section “Create Filters for Updating the Routing Table” in the “Configuring Novell IPX” chapter in the *Network Protocols Configuration Guide, Part 2*.

Enable SLIP and PPP on Virtual Asynchronous Interfaces

The Cisco IOS software permits you to configure asynchronous protocol features, such as SLIP and PPP, on virtual terminal (VTY) lines. SLIP and PPP normally function only on asynchronous interfaces, and not on VTY lines. When you configure a VTY line to support asynchronous protocol features, you are creating *virtual asynchronous interfaces* on the VTY lines. One practical benefit of

virtual asynchronous interfaces is the ability to tunnel SLIP and PPP across X.25, TCP, or LAT networks on VTY lines. You tunnel SLIP and PPP using the protocol translation facility. For more information, refer to the chapter “Configuring Protocol Translation” in this publication.

Perform the tasks in the following sections to configure and use virtual asynchronous interfaces. The first task is required; the remaining tasks are optional.

- Create Virtual Asynchronous Interfaces
- Enable Protocol Translation of SLIP and PPP on Virtual Asynchronous Interfaces
- Enable Dynamic Routing on Virtual Asynchronous Interfaces
- Enable TCP/IP Header Compression on Virtual Asynchronous Interfaces
- Enable Keepalive Updates on Virtual Asynchronous Interfaces
- Set an MTU on Virtual Asynchronous Interfaces
- Enable PPP Authentication on Virtual Asynchronous Interfaces
- Enable PPP Authentication via TACACS on Virtual Asynchronous Interfaces

Note These tasks enable SLIP and PPP on a virtual asynchronous interfaces on a global basis on the router. To configure SLIP or PPP on a per-VTY basis, use the **translate** command.

Create Virtual Asynchronous Interfaces

To create a virtual asynchronous interface, perform the following task in global configuration mode:

Task	Command
Configure all virtual terminal lines to support asynchronous protocol features.	vty-async

Enable Protocol Translation of SLIP and PPP on Virtual Asynchronous Interfaces

One practical benefit of enabling virtual asynchronous interfaces is the ability to tunnel SLIP and PPP over X.25, thus extending remote node capability into the X.25 area. You can also tunnel SLIP and PPP over Telnet or LAT on virtual terminal lines. You can tunnel SLIP and PPP over X.25, LAT, or Telnet, but you do so by using the protocol translation feature in the Cisco IOS software. Refer to the “Configuring Protocol Translation” chapter in this publication for more information about protocol translation.

To tunnel incoming dial-up SLIP or PPP connections over X.25, LAT, or TCP to an IP network, you can use one-step protocol translation or two-step protocol translation, as follows:

- If you are tunneling SLIP or PPP using the one-step method, you do not need to enter the **vty-async** command. Using the **translate** command with the SLIP or PPP keywords for one-step connections automatically enables asynchronous protocol functions on a per-VTY basis. For more information about tunneling SLIP and PPP using protocol translation, refer to the “Configuring Protocol Translation” chapter in this publication. For more information about using the **translate** command with the SLIP or PPP keywords, refer to the “Protocol Translation Configuration Commands” chapter in the *Access Services Command Reference*.

- If you are tunneling SLIP or PPP using the two-step method, you must first enter the **vty-async** command on a global basis. Next, you perform a two-step connection process. For more information about two-step connections, refer to the chapter “Making Connections to Network Devices” in this manual.

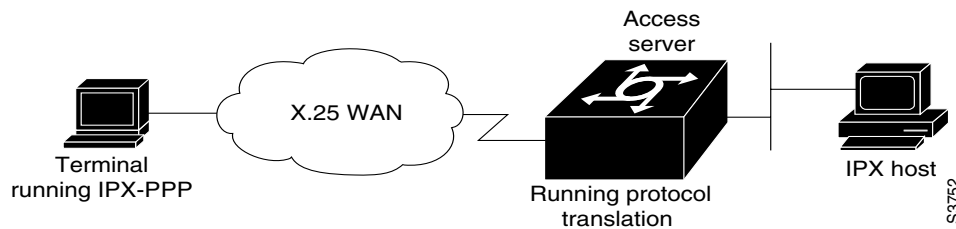
For an example of tunneling SLIP across an X.25 PAD WAN to an IP network, refer to the “Configuring Protocol Translation” chapter in this publication.

Enable IPX–PPP on Virtual Asynchronous Interfaces

You can enable IPX–PPP on virtual terminal lines (VTYs), which permits clients to log into a VTY on a router, invoke a PPP session at the EXEC prompt to a host, and run IPX to the host.

For example, in Figure 30, the client Terminal1 on the X.25 network logs into the VTY on Access Server1, which is configured for IPX–PPP. When the user connects to the access server and the EXEC prompt appears, the user issues the PPP command to connect to the IPX host. The VTY is configured to run IPX, so when the PPP session is established from the access server, Terminal1 can access the IPX host using an IPX application.

Figure 30 IPX–PPP on a Virtual Asynchronous Interface



To enable IPX to run over your PPP sessions on VTY lines, perform the following tasks, beginning in global configuration mode:

Task	Command
Step 1 Enable IPX routing.	ipx routing [<i>node</i>]
Step 2 Create a loopback interface.	interface loopback <i>number</i>
Step 3 Enable a virtual IPX network on the loopback interface.	ipx network <i>network</i> ¹
Step 4 Enable IPX–PPP on VTY lines by assigning the VTY to the loopback interface configured for IPX.	vty-async ipx ppp-client <i>Loopback number</i>

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

Enable Dynamic Routing on Virtual Asynchronous Interfaces

To route IP packets using the IGRP, RIP, and OSPF routing protocols on virtual asynchronous interfaces, perform the following task in global configuration mode:

Task	Command
Enable dynamic routing of IP packets on all virtual terminal lines.	vty-async dynamic-routing

When you make a connection, you must specify the **routing** keyword on the SLIP or PPP command line.

Note The **vty-async dynamic routing** command is similar to the **async dynamic routing** command, except that the **async dynamic routing** command is used for physical asynchronous interfaces, and the **vty-async dynamic-routing** command is used on virtual terminal lines configured for asynchronous protocol functionality.

Enable TCP/IP Header Compression on Virtual Asynchronous Interfaces

You can compress the headers on TCP/IP packets on virtual asynchronous interfaces to reduce their size and increase performance. This feature only compresses 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 virtual asynchronous interfaces using SLIP and PPP encapsulation. You must enable compression on both ends of the connection.

You can optionally specify outgoing packets to be compressed only if TCP incoming packets on the same virtual terminal line are compressed. If you do not specify this option, the Cisco IOS software will compress all traffic. The default is no compression. This option is valid for SLIP.

To compress the headers of outgoing TCP packets on virtual asynchronous interfaces, perform the following task in global configuration mode:

Task	Command
Enable header compression on IP packets on all virtual terminal lines.	vty-async header-compression [passive]

Enable Keepalive Updates on Virtual Asynchronous Interfaces

Keepalive updates are enabled on all virtual asynchronous interfaces by default. To change the keepalive timer or disable it on virtual asynchronous interfaces, perform the following task in global configuration mode:

Task	Command
Specify the frequency with which the Cisco IOS software sends keepalive messages to the other end of an asynchronous serial link.	vty-async keepalive [seconds]

The default interval is 10 seconds. It is adjustable in one-second increments from 0 to 32,767 seconds. To turn off keepalive updates, set the value to 0. A connection is declared down after three update intervals have passed without receiving a keepalive packet.

Virtual terminal lines have very low bandwidth. When adjusting the keepalive timer, large packets can delay the smaller keepalive packets long enough to cause the session to disconnect. You might need to experiment to determine the best value.

Set an MTU on Virtual Asynchronous Interfaces

The maximum transmission unit (MTU) refers to the size of an IP packet. You might want to change to a smaller MTU size for IP packets transmitted on a virtual asynchronous interface for any of the following reasons:

- The SLIP or PPP application at the other end only supports packets up to a certain size.
- You want to ensure a shorter delay by using smaller packets.
- The host Telnet echoing takes longer than 0.2 seconds.

For example, at 9600 baud a 1500 byte packet takes about 1.5 seconds to transmit. This delay would indicate that you want an MTU size of about 200 ($1.5 \text{ seconds} / 0.2 \text{ seconds} = 7.5$ and $1500 \text{ byte packet} / 7.5 = 200 \text{ byte packet}$).

To specify the maximum IP packet size, perform the following task in interface configuration mode:

Task	Command
Specify the size of the largest IP packet that the virtual asynchronous interface can support.	<code>vty-async mtu bytes</code>

The default MTU size is 1500 bytes. Possible values are 64 bytes to 1,000,000 bytes.

The TCP protocol running on the remote device can have a different MTU size than the MTU size configured on your router. Because the Cisco IOS software performs IP fragmentation of packets larger than the specified MTU. Do not change the MTU size unless the SLIP or PPP implementation running on the host at the other end of the asynchronous line supports reassembly of IP fragments.

Enable PPP Authentication on Virtual Asynchronous Interfaces

You can enable Challenge Handshake Authentication Protocol (CHAP) or Password Authentication Protocol (PAP) for authentication of PPP on VTY lines set up for asynchronous protocol features.

Note Passwords cannot contain spaces or underscores. A user with a password containing spaces or underscores will not be able to log into a TTY or VTY line.

Enable CHAP

Access control using Challenge Handshake Authentication Protocol (CHAP) is available on all virtual asynchronous interfaces configured for PPP encapsulation. The authentication feature reduces the risk of security violations on your router.

When CHAP is enabled, a remote device (such as a PC, workstation, or router) attempting to connect to the local router is requested, or “challenged,” to respond.

The challenge consists of an ID, a random number, and either the host name of the local router or the name of the user on the remote device. This challenge is transmitted to the remote device.

The required response consists of two parts:

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

When the local router receives the challenge response, it verifies the secret by looking up the name given in the response and performing the same encryption operation. The secret passwords must be identical on the remote device and the local router.

By transmitting this response, the secret is never transmitted, thus 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 does not request a password during the rest of the session. (The local router can, however, respond to such requests from other devices during a session.)

To use CHAP on virtual asynchronous interfaces for PPP, perform the following task in global configuration mode:

Task	Command
Enable CHAP on all virtual asynchronous interfaces.	vty-async ppp authentication chap

CHAP is specified in RFC 1334. It is an additional authentication phase of the PPP Link Control Protocol.

Once you have enabled CHAP, the local router requires a response from the remote devices. If the remote device does not support CHAP, no traffic is passed to that device.

Enable PAP

Access control using the Password Authentication Protocol (PAP) is available on all virtual asynchronous interfaces configured for PPP encapsulation. The authentication feature reduces the risk of security violations on your router.

To use PAP, perform the following task in interface configuration mode:

Task	Command
Enable PAP on all virtual asynchronous interfaces.	vty-async ppp authentication pap

Enable PPP Authentication via TACACS on Virtual Asynchronous Interfaces

Access control using the Terminal Access Controller Access Control System (TACACS) is available on all virtual asynchronous interfaces configured for PPP encapsulation. The authentication feature reduces the risk of security violations on your router.

To use TACACS with either CHAP or PAP, perform the following task in global configuration mode:

Task	Command
Enable TACACS on all virtual asynchronous interfaces.	vty-async ppp use-tacacs

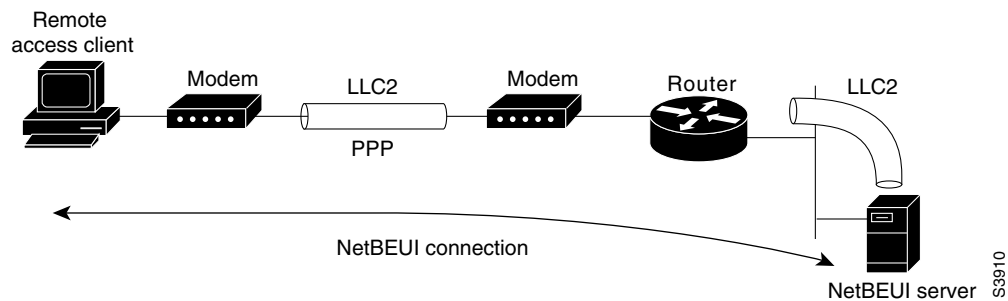
Configure Remote Access to NetBEUI Services

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 IBM's original Network Basic Input/Output System (NetBIOS). 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. For this release, 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. Refer to Figure 31. An 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 31 NetBEUI Connection



By supporting NetBEUI remote clients over PPP, Cisco routers function as a native NetBEUI dialin 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, you must configure interfaces on the remote access client side and the remote router side. Perform the following task, beginning in interface configuration mode:

Task	Command
Enable NetBEUI's NetBIOS Frames Protocol on each side of a NetBEUI connection.	netbios nbf

To view NetBEUI connection information, perform the following task in EXEC mode:

Task	Command
View NetBEUI connection information.	show nbf sessions

Configure Performance Parameters

To tune IP performance, complete the tasks in the following sections:

- Compress TCP Packet Headers

- Set the TCP Connection Attempt Time
- Compress IPX Packet Headers over PPP
- Enable Fast Switching
- Control Route Cache Invalidation

Compress TCP Packet Headers

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 only compresses 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 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 if 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.

To enable compression, perform either of the following optional tasks in interface configuration mode:

Task	Command
Enable TCP header compression.	<code>ip tcp header-compression [passive]</code>
Specify the total number of header compression connections that can exist on an interface.	<code>ip tcp compression-connections <i>number</i></code> ¹

1. This command is documented in the “IP Commands” chapter of the *Network Protocols Command Reference, Part 1*.

Note When compression is enabled, fast switching is disabled. Fast processors can handle several fast interfaces, such as T1s, that are running header compression. However, you should think carefully about your network’s traffic characteristics before compressing TCP headers. You might want to use the monitoring commands to help compare network utilization before and after enabling header compression.

Set the 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 do so. This amount of time is not sufficient in networks that have dial-up asynchronous connections, such as a network consisting of dial-on-demand links that are implemented over modems, because it will affect your ability to 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, perform the following task in global configuration mode:

Task	Command
Set the amount of time the Cisco IOS software will wait to attempt to establish a TCP connection.	ip tcp synwait-time <i>seconds</i> ¹

1. This command is documented in the “IP Commands” chapter of the *Network Protocols Command Reference, Part 1*.

Compress IPX Packet Headers 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, known also 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’s 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 in size. 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.

To configure CIPX, perform the following task in global configuration mode:

Task	Command
Compress IPX packet headers in a PPP session.	ipx compression cipx <i>number-of-slots</i>

Note It is recommended 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.

Enable 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.

To enable or disable fast switching, perform the following tasks in interface configuration mode:

Task	Command
Enable fast-switching (use of a high-speed route cache for IP routing).	ip route-cache ¹
Disable fast switching and enable load balancing on a per-packet basis.	no ip route-cache ¹

1. These commands are documented in the “IP Commands” chapter of the *Network Protocols Command Reference, Part 1*.

Control 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.

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

Task	Command
Allow immediate invalidation of the cache.	no ip cache-invalidate-delay ¹
Delay invalidation of the cache.	ip cache-invalidate-delay [<i>minimum maximum quiet threshold</i>] ¹

1. These commands are documented in the “IP Commands” chapter of the *Network Protocols Command Reference, Part 1*.

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.

Optimize Available Bandwidth

Asynchronous lines have relatively low bandwidth and can easily be overloaded, resulting in slow traffic across these lines.

To optimize available bandwidth, perform any of the following tasks:

- Configure Header Compression
- Force Header Compression at the EXEC Level

Configure Header Compression

One way to optimize available bandwidth is by using TCP header compression. Van Jacobson TCP header compression (defined by RFC 1144) can increase bandwidth availability between two and five times when compared to lines not using header compression. Theoretically, it can improve bandwidth availability by a ratio of seven to one.

To configure header compression, perform the following task in interface configuration mode:

Task	Command
Configure Van Jacobson TCP header compression on the asynchronous link.	ip tcp header-compression [on off passive]

Force Header Compression at the EXEC Level

On SLIP interfaces, you can force header compression at the EXEC prompt on a line on which header compression has been set to passive. This allows more efficient use of the available bandwidth and does not require entering privileged configuration mode.

To implement header compression, perform the following task in interface configuration mode:

Task	Command
Allow status of header compression to be assigned at the user level.	ip tcp header compression passive

For PPP interfaces, the **passive** option functions the same as the **on** option.

See the “Making Connections to Network Devices” chapter for information about the **slip** and **ppp** EXEC commands. You cannot force header compression if header compression on the asynchronous interface is off.

Specify the MTU Size of IP Packets

The maximum transmission unit (MTU) refers to the size of an IP packet. You might want to change to a smaller MTU size for any of the following reasons:

- The SLIP or PPP application at the other end only supports packets up to a certain size.
- You want to ensure a shorter delay by using smaller packets.
- The host Telnet echoing takes longer than 0.2 seconds.

For example, at 9600 baud a 1500 byte packet takes about 1.5 seconds to transmit. This delay would indicate that you want an MTU size of about 200 ($1.5 \text{ seconds} / 0.2 \text{ seconds} = 7.5$ and $1500 \text{ byte packet} / 7.5 = 200 \text{ byte packet}$).

To specify maximum IP packet size, perform the following task in interface configuration mode:

Task	Command
Specify the size of the largest IP packet that the asynchronous line can support.	ip mtu <i>bytes</i>

The MTU size can be negotiated by TCP, regardless of the asynchronous interface settings. In other words, TCP running on the device to which the router is connected can negotiate for a different MTU size than is configured on the router. The Cisco IOS software performs IP fragmentation of packets larger than the specified MTU. Do not change the MTU size unless the SLIP or PPP implementation running on the host at the other end of the asynchronous line supports reassembly of IP fragments. Because each fragment occupies a spot in the output queue, it might also be necessary to increase the size of the SLIP or PPP hold queue, if your MTU size has allowed a high amount of packet fragmentation in the output queue.

Provide Backward Compatibility for SLIP and PPP

To provide backward compatibility for client software scripts expecting SLIP and PPP dialog to be formatted with software release 9.1 or earlier, use the **service old-slip-prompts** global configuration command. You can format SLIP and PPP transmission by performing the following task in global configuration mode.

Task	Command
Format SLIP and PPP dialogs.	service old-slip-prompts

Modify the IP Output Queue Size

The IP output queue stores packets received from the network that are waiting to be sent to the asynchronous client. You can limit the size of the IP output queue to enhance performance by performing the following task in interface configuration mode:

Task	Command
Change the size of the IP output hold queue.	hold-queue packets

Specify IP Access Lists

Access lists allow the system administrator to control the hosts that a PC can access through a router. Separate access lists can be defined for asynchronous and for other connections.

This section describes the following tasks:

- Define access control on packets from the IP host
- Define access control on packets to the IP host

Refer to the “Configuring IP” chapter in the *Network Protocols Configuration Guide, Part 1* for information about defining IP access lists.

To define an access list for packets from the IP host, perform the following task in interface configuration mode:

Task	Command
Configure an access list for packets <i>from</i> the IP host.	ip access-group access-list-number in

To define an access list for packets to the IP host, perform the following task in interface configuration mode:

Task	Command
Configure an access list for packets being sent <i>to</i> the IP host.	ip access-group access-list-number out

Configure Support for Extended BOOTP Requests

To configure Cisco IOS software to respond to BOOTP requests from client machines, perform the following task in global configuration mode:

Task	Command
Specify the router network information that is sent in response to BOOTP requests.	async-bootp <i>tag</i> [: <i>hostname</i>] <i>data</i>

Monitor and Maintain Asynchronous Interfaces

This section describes the following monitoring and maintenance tasks that you can perform on asynchronous interfaces:

- Monitor and maintain asynchronous activity
- Debug asynchronous interfaces
- Debug PPP

To monitor and maintain asynchronous activity, perform one or more of the following tasks in privileged EXEC mode:

Task	Command
Return a line to its idle state.	clear line <i>line-number</i>
Display parameters that have been set for extended BOOTP requests.	show async bootp
Display statistics for asynchronous interface activity.	show async status
Display the status of asynchronous line connections.	show line [<i>line-number</i>]

To debug asynchronous interfaces, perform the following task in privileged EXEC mode:

Task	Command
Displays errors, changes in interface state, and log input and output.	debug async { framing state packets }

To debug PPP links, perform the following tasks in privileged EXEC mode:

Task	Command
Enable debugging of PPP protocol negotiation process.	debug ppp negotiation
Display PPP protocol errors.	debug ppp error
Display PPP packets sent and received.	debug ppp packet
Display errors encountered during remote or local system authentication.	debug ppp chap ¹

1. Refer to the chapter “Configuring Dial-on-Demand Routing” in the *Wide-Area Networking Configuration Guide* for more information about the Challenge Handshake Authentication Protocol (CHAP).

Asynchronous Interface Configuration Examples

This section contains asynchronous configuration examples. Each configuration is designed to illustrate different communication requirements.

- Dedicated Asynchronous Interface Configuration Example
- IP–SLIP—Asynchronous Interface Example
- AppleTalk–PPP Example
- IP–PPP Example
- IPX–PPP—Loopback Interface Example
- IPX–PPP—Using Dedicated IPX Network Numbers for Each Interface Example
- IPX–PPP over X.25 to an IPX Network on VTY Lines Example
- Restricted Access on an Asynchronous Interface Example
- Asynchronous Routing and Dynamic Addressing Configuration Example
- TCP Header Compression Configuration Example
- Conserving Network Addresses Using the IP Unnumbered Feature Example
- Configuring Routing on a Dedicated Dial-In Router Example
- Configuring an Asynchronous Interface as the Only Network Interface Example
- Configuring IGRP Example
- Configuring an Interface Example
- Remote Network Access Using PPP—A Basic Configuration Example
- Remote Network Access Using PPP—Routing IP Example
- Remote Network Access—Leased Line with Dial-Backup Using PPP Example

Dedicated Asynchronous Interface Configuration Example

The following example assigns an IP address to an asynchronous interface and places the line in dedicated network mode. Setting the stop bit to 1 is a performance enhancement.

```
line 20
  location Department PC Lab
  stopbits 1
  speed 19200
!
interface async 20
  async default ip address 182.32.7.51
  async mode dedicated
```

IP–SLIP—Asynchronous Interface Example

The following example configures IP–SLIP on asynchronous interface 6. The IP address for the interface is assigned to Ethernet 0, interactive mode has been enabled, and the IP address of the client PC running SLIP has been specified.

IP and the appropriate IP routing protocols have already been enabled on the server.

```
Router(config)# interface async 6
Router(config-if)# ip unnumbered ethernet 0
Router(config-if)# encapsulation slip
```

```
Router(config-if)# async mode interactive
Router(config-if)# async default ip address 172.18.1.128
```

AppleTalk–PPP Example

The following example configures asynchronous interface 4 on the router so that users can access AppleTalk zones by dialing into the router via PPP to 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. Routing is not supported on the asynchronous interface 6.

```
Router(config)# interface async 6
Router(config-if)# encapsulation ppp
Router(config-if)# appletalk virtual-net 12345 saivite
Router(config-if)# appletalk client-mode
```

IP–PPP Example

The following example configures IP–PPP on asynchronous interface 6. The IP address for the interface is assigned to Ethernet 0, interactive mode has been enabled, and the IP address of the client PC running PPP has been specified.

IP and the appropriate IP routing protocols have already been enabled on the server.

```
Router(config)# interface async 6
Router(config-if)# ip unnumbered ethernet 0
Router(config-if)# encapsulation ppp
Router(config-if)# async mode interactive
Router(config-if)# peer default ip address 172.18.1.128
```

IPX–PPP—Loopback Interface Example

The following example shows the process of configuring IPX to run over PPP on an asynchronous interface. The asynchronous interface is associated with a loopback interface configured to run IPX. This example enables a non-routing IPX client to connect to the router.

```
Router(config)# ipx routing 0000.0c07.b509
Router(config)# interface loopback0
Router(config-if)# no ip address
Router(config-if)# ipx network 544
Router(config-if)# ipx sap-interval 2000
Router(config-if)# exit
Router(config)# interface ethernet0
Router(config-if)# ip address 172.21.14.64
Router(config-if)# ipx network AC150E00
Router(config-if)# ipx encapsulation SAP
Router(config-if)# exit
Router(config)# interface async 3
Router(config-if)# ip unnumbered ethernet0
Router(config-if)# encapsulation ppp
Router(config-if)# async mode interactive
Router(config-if)# async default ip address 172.18.1.128
Router(config-if)# ipx ppp-client loopback0
Router(config-if)# ipx sap-interval 0
```

In this example, IPX client connections are permitted to asynchronous interface 3, which is associated with loopback interface 0. Loopback interface 0 is configured to run IPX. Routing updates have been filtered on asynchronous interface 3. Routing updates take up a great deal of bandwidth, and asynchronous interfaces have low bandwidth.

IPX-PPP—Using Dedicated IPX Network Numbers for Each Interface Example

The following example shows the process of configuring IPX to run over PPP on an asynchronous interface. A dedicated IPX network number has been specified for each interface, which can require a substantial number of network numbers for a large number of interfaces. This example permits an IPX client with routing enabled to connect with the router.

```
Router(config)# ipx routing 0000.0c07.b509
Router(config)# interface async 6
Router(config-if)# ip unnumbered ethernet0
Router(config-if)# encapsulation ppp
Router(config-if)# async mode interactive
Router(config-if)# ipx network AC150E00
Router(config-if)# ipx sap-interval 0
```

In this example, IPX client connections are permitted to asynchronous interface 6, which has a unique IPX network number. Routing updates have been filtered on asynchronous interface 6. Routing updates take up a great deal of bandwidth, and asynchronous interfaces have low bandwidth.

IPX-PPP over X.25 to an IPX Network on VTY Lines Example

The following example shows the process of enabling IPX-PPP on VTY lines. First, you enable PPP to run on VTY lines, then you associate the VTY line with a loopback interface configured to run IPX. This example enables a non-routing IPX client to connect to the router.

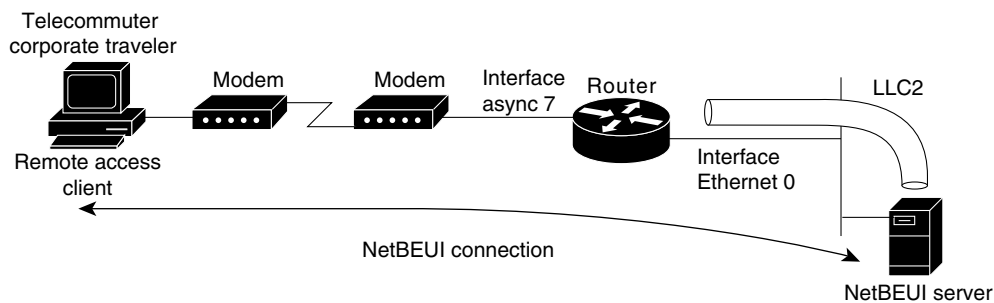
```
Router(config)# ipx routing 0000.0c07.b509
Router(config)# interface loopback0
Router(config-if)# no ip address
Router(config-if)# ipx network 544
Router(config-if)# exit
Router(config-if)# vty-async ipx ppp-client loopback0
```

In this example, IPX client connections are permitted to VTY lines, which have been associated with loopback interface 0. Loopback interface 0 is configured with an IPX network number that is used by the VTY lines.

Remote Node NetBEUI Example

In the following example, asynchronous interface 7 and ethernet interface 0 are configured to enable NetBEUI connectivity between the corporate telecommuter's client and the remote access (NetBEUI) server. The PC client is running a legacy application—Chat—in Windows NT, to connect with the remote server. Refer to Figure 32.

Figure 32 Connecting a Remote NetBEUI Client to a Server through a Router



S3911

Configuration for Router:

```
2511(config)#interface async 7
2511(config-if)#netbios nbf
2511(config-if)#encapsulation ppp
```

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

Restricted Access on an Asynchronous Interface Example

The following example assumes that users are restricted to certain servers designated as asynchronous servers, but that normal terminal users can access anything on the local network.

```
! access list for normal connections
access-list 1 permit 172.16.0.0 0.0.255.255
!
access-list 2 permit 172.16.42.55
access-list 2 permit 172.16.111.1
access-list 2 permit 172.16.55.99
!
interface async 6
  async dynamic address
  ip access-group 1 out
  ip access-group 2 in
```

Asynchronous Routing and Dynamic Addressing Configuration Example

The following example shows a simple configuration that allows routing and dynamic addressing. With this configuration, if the user specifies **/routing** in the EXEC **slip** or **ppp** command, routing protocols will be sent and received.

```
interface async 6
  async dynamic routing
  async dynamic address
```

TCP Header Compression Configuration Example

The following example configures async interface 7 with a default IP address, allowing header compression if it is specified in the **slip** or **ppp** connection command entered by the user or if the connecting system sends compressed packets.

```
interface async 7
  ip address 172.31.79.1
  async default ip address 172.31.79.2
  ip tcp header-compression passive
```

Conserving Network Addresses Using the IP Unnumbered Feature Example

The following example shows how to configure your router for routing using unnumbered interfaces. The source (local) address is shared between Ethernet 0 and async 6 (172.18.1.1). The default remote address is 172.18.1.2.

```
interface ethernet 0
  ip address 172.18.1.1 255.255.255.0
!
interface async 6
  ip unnumbered ethernet 0
  async dynamic routing
```

```
! default address is on the local subnet
async dynamic address
async default ip address 172.18.1.2
ip tcp header-compression passive
```

The following example shows how the IP unnumbered configuration works. Although the user assigned an address, the system response shows the interface as unnumbered, and the address typed by the user will be used only in response to BOOTP requests.

```
Router> slip /compressed 10.11.11.254
Password:
Entering async mode.
Interface IP address is unnumbered, MTU is 1500 bytes.
Header compression is On.
```

Configuring Routing on a Dedicated Dial-In Router Example

In the following example, the router is set up as a dedicated dial-in router. Interfaces are configured as IP unnumbered to conserve network resources, primarily IP addresses.

```
ip routing
interface ether 0
  ip address 10.129.128.2 255.255.255.0
!
interface async 1
  ip unnumbered ethernet 0
  async dynamic routing
! The addresses assigned with SLIP or PPP EXEC commands are not used except
! to reply to BOOTP requests.
! Normally, the routers dialing in will have their own address
! and not use BOOTP at all.
  async default ip address 10.11.11.254
!
interface async 2
  ip unnumbered ethernet 0
  async default ip address 10.11.12.16
  ip tcp header-compression passive
  async mode dedicated
!
! run RIP on the asynchronous lines, because few implementations of SLIP
! understand IGRP. Run IGRP on the ethernet (and in the local network).
!
router igrp 110
  network 10.11.12.0
! send routes from the asynchronous lines on the production network.
  redistribute RIP
! don't send IGRP updates on the async interfaces
  passive-interface async 1
!
  router RIP
  network 10.11.12.0
  redistribute igrp
  passive-interface ethernet 0
! consider filtering everything except a default route from the routing
! updates sent on the (slow) asynchronous lines
  distribute-list 1 out
  ip unnumbered async 2
  async dynamic routing
```

Configuring an Asynchronous Interface as the Only Network Interface Example

In the following example, one of the asynchronous lines is used as the only network interface. The router is used primarily as a terminal server, but is at a remote location and dials into the central site for its only network connection.

```
ip default-gateway 10.11.12.2
interface ethernet 0
 shutdown
interface async 1
 async dynamic routing
 ip tcp header-compression on
 async default ip address 10.11.16.12
 async mode dedicated
 ip address 10.11.12.32 255.255.255.0
```

Configuring IGRP Example

In the following example, only the IGRP TCP/IP routing protocol is running; it is assumed that the systems that are dialing in to use routing will either support IGRP or have some other method (for example, a static default route) of determining that the router is the best place to send most of its packets.

```
router igrp 111
 network 10.11.12.0
interface ethernet 0
 ip address 10.11.12.92 255.255.255.0
!
interface async 1
 async default ip address 10.11.12.96
 async dynamic routing
 ip tcp header-compression passive
 ip unnumbered ethernet 0

line 1
 modem ri-is-cd
```

Configuring an Interface Example

The following configuration shows interface and line configuration. The interface is configured with access lists, passive header compression and a default address. The line is configured for TACACS authentication.

```
interface async 1
 ip access-group 1 in
 ip access-group 1 out
 ip tcp header-compression passive
 async default ip address 172.31.176.201

line 1
 login tacacs
 location 457-5xxx
 exec-timeout 20 0
 password XXXXXXXX
 session-timeout 20
 stopbits 1
```

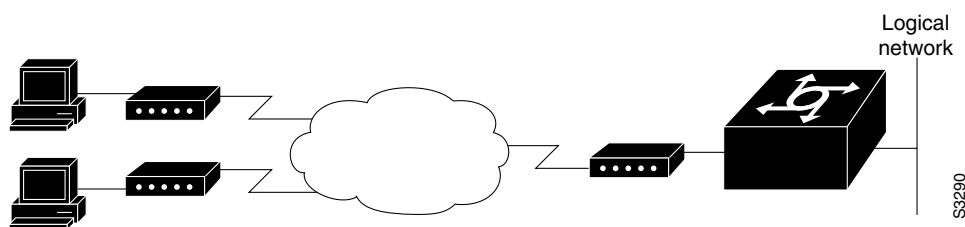
Remote Network Access Using PPP—A Basic Configuration Example

Figure 33 illustrates a simple network configuration comprised of 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 configuration you need to configure the following:

- 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 33 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 typing PPP 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
    
```

Remote Network Access Using PPP—Routing IP Example

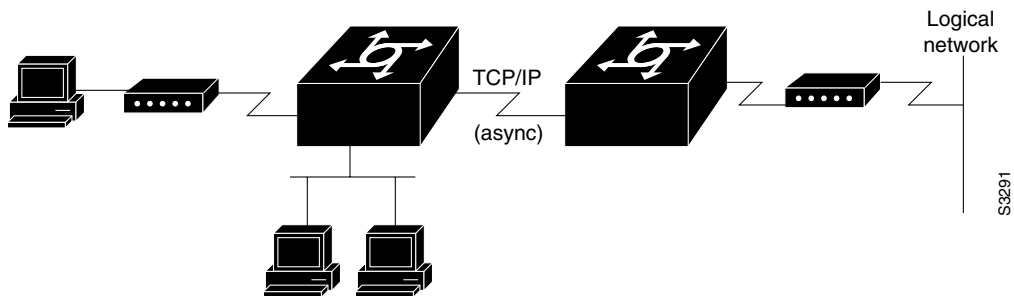
Figure 34 illustrates a network configuration that provides routing functionality, allowing routing updates to be passed across the asynchronous lines.

This network is comprised 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 34 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, issue the following commands:

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

Next, complete these steps to configure the asynchronous lines between the access servers, starting 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 *Network Protocols Configuration Guide, Part 1* using one of the following methods. The server can route packets three different ways:

- 1 Use ARP, which is the default behavior.
- 2 Use a default-gateway by issuing the command **ip default-gateway** *x.x.x.x*, where *x.x.x.x* is the IP address of a locally attached router.

- 3 Run an IP routing protocol (RIP, IGRP, EIGRP, or OSPF).

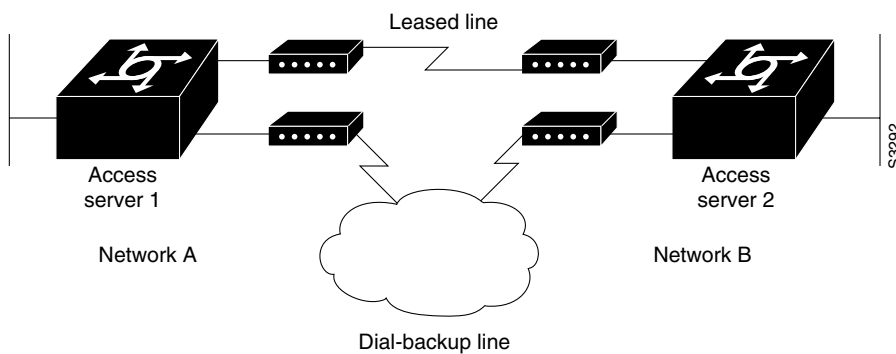
Remote Network Access—Leased Line with Dial-Backup Using PPP Example

Figure 35 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 public switched telephone network 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.

In this scenario, you will 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 access server’s related network

Figure 35 Asynchronous Leased Line with Backup



The configuration is as 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
```

