

Configuring Asynchronous PPP and SLIP

This chapter describes how to configure PPP or SLIP encapsulation over asynchronous lines for connection-oriented protocols such as IP, IPX, and AppleTalk. For a complete description of the PPP and SLIP commands in this chapter, refer to the “Asynchronous PPP and SLIP Commands” chapter in the *Dial Solutions Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

Cisco’s Implementation of PPP and SLIP

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. PPP and SLIP over asynchronous dial-up modems allow a home computer to be connected to a network without the cost of a leased line. Dial-up 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.

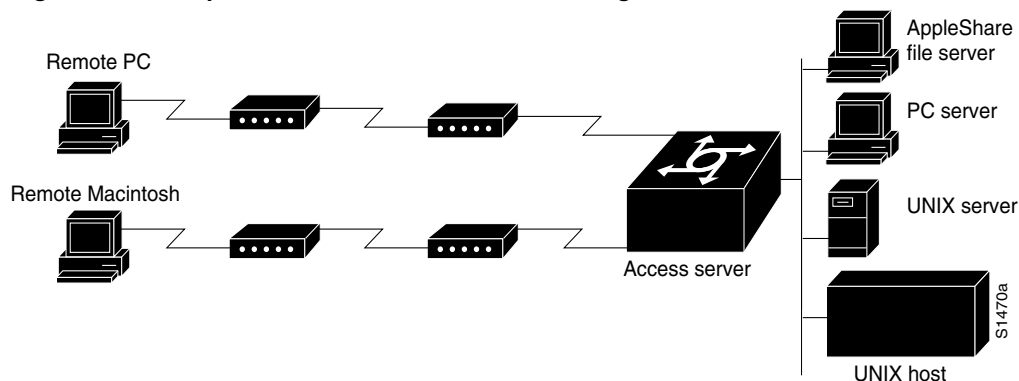
The Cisco IOS software concentrates a large number of SLIP or PPP PC or workstation client hosts onto a network interface that allows the PCs to communicate with any host on the network. The Cisco IOS software can support any combination of SLIP or PPP lines and lines dedicated to normal asynchronous devices such as terminals and modems. 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 91 illustrates a typical asynchronous SLIP or PPP remote-node configuration.

Figure 91 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.

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 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 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 *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.

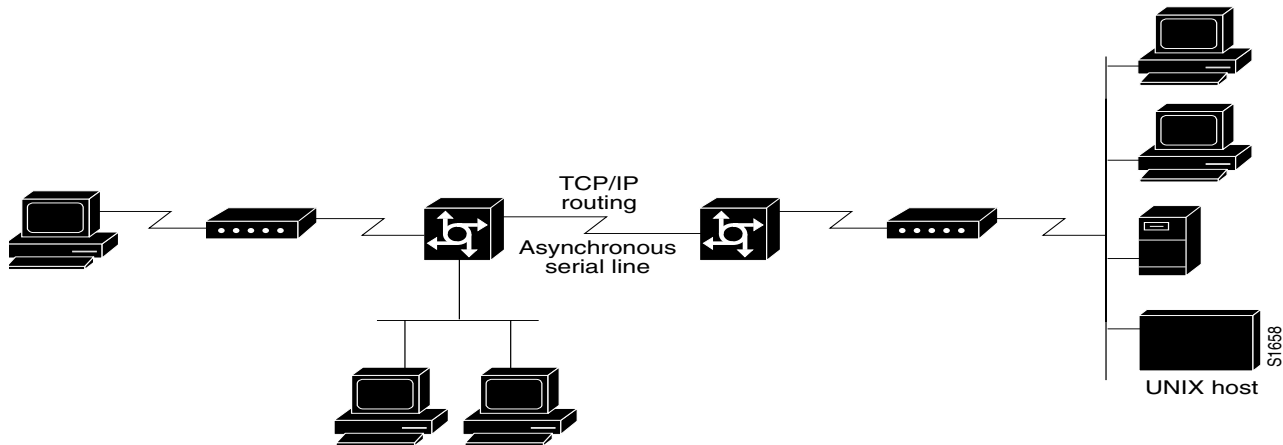
The Cisco IOS software also supports IP routing connections for communication that requires connecting one network to another.

The Cisco IOS software supports protocol translation for PPP and SLIP 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 and Virtual Asynchronous Devices" 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 92 shows a sample asynchronous routing configuration.

Figure 92 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.

Asynchronous PPP and SLIP Task List

The following tasks are provided:

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

Configure Network-Layer Protocols over PPP and SLIP

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

Configure IP–PPP

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

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

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 over X.25 to an IPX Network on vty Lines” later in this section.

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:

Step	Command	Purpose
1	ipx routing [<i>node</i>]	Enable IPX routing.
2	interface loopback <i>number</i>	Create a loopback interface, which is a virtual interface, existing only inside the router.
3	ipx network <i>network</i> ¹	Enable IPX routing on the loopback interface.
4	exit	Exit to global configuration mode.
5	interface async <i>number</i>	Enter interface configuration mode for the asynchronous interface.
6	ip unnumbered <i>type number</i>	Configure IP unnumbered routing on the interface.
7	encapsulation ppp	Enable PPP encapsulation on the interface.
8	async mode interactive	Enable interactive mode on an asynchronous interface.
9	ipx ppp-client Loopback <i>number</i>	Assign the asynchronous interface to the loopback interface configured for IPX.
10	ipx sap-interval 0	Turn off SAP updates to optimize bandwidth on asynchronous interfaces.

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. 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:

Step	Command	Purpose
1	ipx routing [<i>node</i>]	Enable IPX routing.
2	interface loopback <i>number</i>	Create a loopback interface, which is a virtual interface, existing only inside the router.
3	encapsulation ppp	Enable PPP encapsulation on the interface.
4	async mode interactive	Enable interactive mode on an asynchronous interface.
5	ipx network <i>network</i> ¹	Enable IPX routing on the interface.
6	ipx sap-interval 0	Turn off SAP updates to optimize bandwidth on asynchronous interfaces.

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

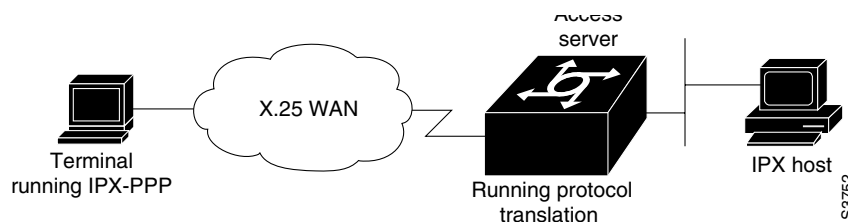
If you are configuring IPX–PPP on asynchronous interfaces, you should filter routing updates on the interface. Most asynchronous serial links have very low bandwidth, and routing updates take up a great deal of bandwidth. To filter routing updates, refer to the section “Create Filters for Updating the Routing Table” in the “Configuring Novell IPX” chapter in the *Network Protocols Configuration Guide, Part 2*.

Enable IPX–PPP over X.25 to an IPX Network on vty Lines

You can enable IPX–PPP on virtual terminal lines (vty), 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 93, the client terminal on the X.25 network logs into the access server via a vty line, 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, the terminal can access the IPX host using an IPX application.

Figure 93 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:

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

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

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:

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

Configure IP–SLIP

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

Step	Command	Purpose
1	ip address <i>ip-address mask</i> or ip unnumbered <i>type number</i>	Step 1 Configure IP routing on the interface. or Configure IP unnumbered routing on a serial interface.
2	encapsulation slip	Step 3 Enable SLIP encapsulation on the serial interface.
4	async mode interactive	Step 5 Enable interactive mode on an asynchronous interface.

Configure 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 into becomes a remote point-of-presence (POP) for the user’s home network. 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 and/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 94 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 94 Asynchronous Host Mobility

Asynchronous host mobility is enabled with the **tunnel EXEC** command and the **ip tcp async-mobility server** global 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 (vty), 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 enable asynchronous host mobility, perform the following tasks in user EXEC mode:

Step	Command	Purpose
1	configure terminal	Enter global configuration mode.
2	ip tcp async-mobility server	Enable asynchronous listening on TCP tunnel port 57.
3	exit	Go to User EXEC mode.
4	tunnel host	Set up a network layer connection to a router by specifying its Internet name or address. Replace the <i>host</i> argument with the name or address of the device you want to connect to.

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.

Make Additional Remote Node Connections

This section describes how to connect devices across telephone lines by using the Point-to-Point Protocol (PPP) and Serial Line Internet Protocol (SLIP).

This section contains the following sections:

- PPP Connections
- SLIP Connections

PPP Connections

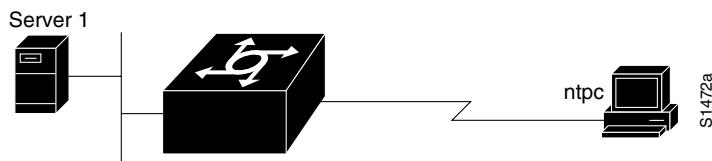
When you connect from a remote node computer, through an asynchronous port on an access server to the EXEC facility, and you want to connect from the access server to a device on the network, perform the following task in EXEC mode:

Command	Purpose
<code>ppp /default { {remote-ip-address remote-name} [@tacacs-server] [/routing]}</code>	Create 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, in Figure 95, if you are working at home on the device `ntpc` and want to connect to Server 1 using PPP, you could dial into the access server. When you connect to the EXEC prompt on the access server, type the `ppp` command to connect with the device.

Figure 95 Using the PPP EXEC Command



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

SLIP Connections

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

Command	Purpose
<code>slip /default { {remote-ip-address remote-name} [@tacacs-server] [/routing]} [/compressed]</code>	Create 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.

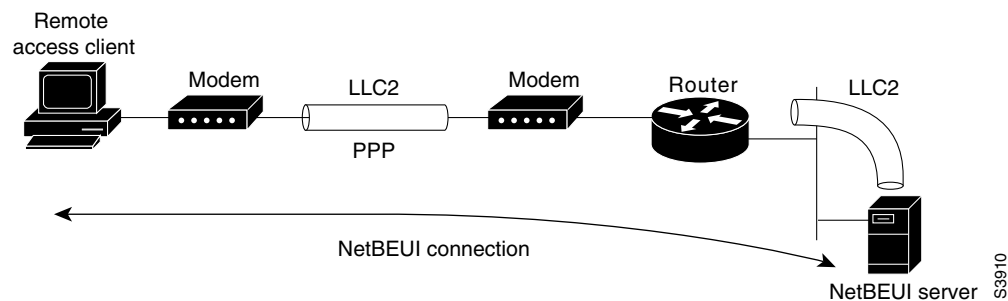
Configure Remote Access to NetBEUI Services

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 96. 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 96 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:

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

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

Command	Purpose
show nbf sessions	View NetBEUI connection information.

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:

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

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:

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

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:

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

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:

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

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:

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

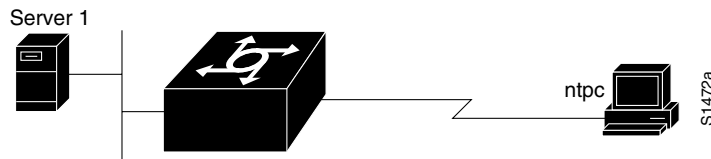
Note This task normally should not be necessary. It should be performed only under the guidance of technical staff. Incorrect configuration can seriously degrade the performance of your router.

PPP and SLIP Connection Examples

The following example shows a line that is in asynchronous mode using PPP encapsulation (see Figure 97). The name of the PC is ntpc. Assuming that the name ntpc is in the Domain Naming System (DNS), the access server can match a real IP address. The PC must be running a terminal emulator program.

```
router> ppp ntpc@server1
```

Figure 97 Using the PPP EXEC Command



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 192.31.7.28, MTU is 1524 bytes
```

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

```
router> slip
IP address or hostname? 192.31.6.15
Password:
Entering SLIP mode
Your IP address is 192.31.6.15, MTU is 1524 bytes
```

In the preceding example, the address 192.31.6.15 has been assigned as the default. Password verification is still required before SLIP mode can be enabled.

```
router> slip default
Password:
Entering SLIP mode
Your IP address is 192.31.6.15, MTU is 1524 bytes
```

The following example illustrates the implementation of header compression on the interface with the IP address 128.66.2.1:

```
router> slip 128.66.2.1 /compressed
Password:
Entering SLIP mode.
Interface IP address is 128.66.2.1, 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 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 1.0.0.1@parlance
Password:
Entering SLIP mode.
Interface IP address is 1.0.0.1, MTU is 1500 bytes
Header compression will match your system.
```

The following example configures multilink PPP 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 200.200.100.9 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
```