CHAPTER 15

Configuring SLIP and PPP

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

Refer to the *Cisco Access Connection Guide* for information about EXEC user commands and establishing SLIP and PPP connections.

Cisco's Implementation of SLIP and PPP

SLIP and PPP define methods of sending Internet Protocol (IP) packets over standard RS-232 asynchronous serial lines with minimum line speeds of 1200 baud.

Using SLIP or PPP encapsulation over asynchronous lines is an inexpensive way of connecting 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 telecommuting or backup connectivity. Both public-domain and vendor-supported SLIP and PPP implementations are available for a variety of computer applications.

The communication server concentrates a large number of SLIP or PPP PC or workstation client hosts onto a network interface allowing the PCs to communicate with any host on the network. The communication server 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.

PPP is a newer, more robust protocol than SLIP and it contains protocols that can detect or prevent misconfiguration. SLIP is an older protocol that is supported on more machines.

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 lines.

Figure 15-1 illustrates a typical asynchronous SLIP or PPP telecommuting configuration.

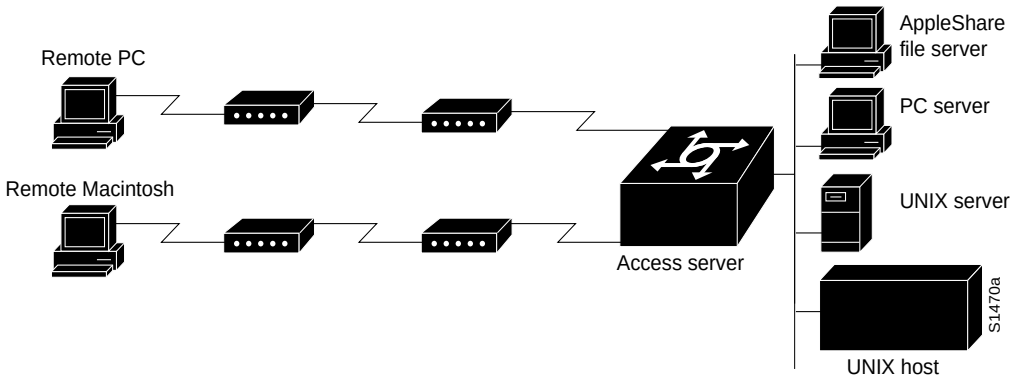


Figure 15-1 Sample SLIP or PPP Telecommuting Configuration

Note SLIP and PPP are not supported over X.25.

Responding to BOOTP Requests

There is an asynchronous BOOTP server in your communication server. This means that SLIP and PPP clients can send BOOTP requests to the communication server, and the communication server will respond 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 communication server can respond with the information.

BOOTP allows a client machine to discover its own IP address, the address of the communication server, 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 bootfile is selected; then the file is transferred, typically using TFTP.

BOOTP compares to RARP as follows: Reverse Address Resolution Protocol (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," later in this publication, for more information about RARP.) However, RARP is a hardware link protocol, so it can only be implemented on hosts that have special kernel or driver modifications that allow access to these raw packets. BOOTP does not require kernel modifications.

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

Asynchronous Network Connections and Routing

Line configuration commands configure a connection to a terminal or a modem. Interface configuration (**async**) commands described in the "Configuring SLIP and PPP" chapter of this publication configure a line as an asynchronous network interface over which networking functions are performed.

Your communication server also supports IP routing connections for communication that requires connecting one network to another.

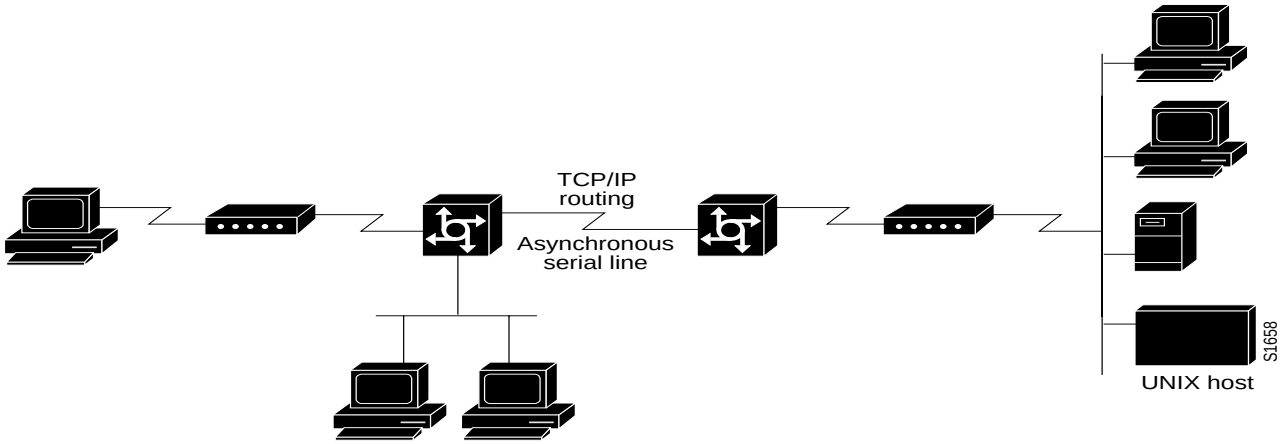
Beginning with IOS Release 10.3, your communication server supports protocol translation for SLIP and PPP between other network devices running Telnet, LAT, or X.25. The IOS software also supports translation for byte-oriented traffic to and from a packet assembler/disassembler (PAD). For example, you can send IP packets across a public X.25 PAD network using SLIP or PPP encapsulation when SLIP or PPPprotocol 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 15-2 shows a sample asynchronous routing configuration.

Note Disable software flow control on SLIP and PPP lines.

Figure 15-2 Sample Asynchronous Routing Configuration



Asynchronous Interfaces and Broadcasts

Communication servers recognize a variety of IP broadcast addresses. When a communication server receives an IP packet from an asynchronous client, it rebroadcasts the packet onto the network without changing the IP header. The communication server does not alter the packet's broadcast address to match the form of broadcast address it prefers.

The communication server receives a copy of asynchronous client broadcasts, and responds to BOOTP requests with the current IP address assigned to the asynchronous interface on which the request was received. This facility allows the asynchronous client software to automatically determine its own IP address.

Telecommuting Configuration Task List

To configure your communication server to support telecommuting, 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
- Enable SLIP and PPP on Virtual Asynchronous Interfaces
- Configure Automatic Protocol Startup
- Configure Performance Parameters
- Optimize Available Bandwidth
- Specify the MTU Size of IP Packets
- Improve Asynchronous PPP Performance
- Modify the IP Output Queue Size
- Specify IP Access Lists
- Configure Support for Extended BOOTP Requests
- Monitor and Maintain Asynchronous Interfaces

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 your communication server to support telecommuting, 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 Device
- Assign IP Addresses for Local Devices
- Enable Asynchronous Routing
- Make SLIP and PPP connections at the EXEC level if you have configured interactive mode. Refer to the *Cisco Access Connection Guide* for more information about making SLIP and PPP connections.

Note In Release 9.1, SLIP was configured and monitored using **slip** line, EXEC, and debug commands. Beginning with Release 9.21, SLIP and PPP asynchronous interfaces are configured using **async** commands in interface command mode.

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** *unit*

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**}

In order to use SLIP or PPP, the communication server must be configured with an IP routing protocol or with the **ip host-routing** command. This configuration is done automatically if you are using old-style **slip address** commands, but you must configure it manually if you configure SLIP or PPP via the **interface async** 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 communication server 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 telecommuting 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**

Refer to the chapter "Managing the System," earlier in this publication, for more information about automatic dialing using DTR.

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 Device

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 communication server 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 user 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.	**async default ip address** *address*

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 or PPP 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 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 need only enter the **slip default** or **ppp default** EXEC command and let the server select the address to use. See the *Cisco Access Connection Guide* 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 the Terminal Access Controller Access System (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 communication server 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 on the communication server, 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 on the communication server whether or not the system on the other end of the asynchronous link supports this feature. 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 on the link.

Enable Asynchronous Routing

To route IP packets, perform the following task in interface configuration mode to enable routing protocols IGRP, RIP, and OSPF, on an interface.

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

When the user makes a connection, they must specify **/routing** on the SLIP or PPP command line.

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 lines. SLIP and PPP normally function only on asynchronous interfaces, and not on virtual terminal lines. When you configure a virtual terminal line to support asynchronous protocol features, you are creating *virtual asynchronous interfaces* on the virtual terminal lines. One practical benefit of virtual asynchronous interfaces is the ability to tunnel SLIP and PPP inside of X.25, TCP, or LAT on virtual terminal 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 communication server. 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 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 or PPP, 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 and Communication Servers 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 protocol translation chapter in the *Cisco Access Connection Guide*.

To make a connection to a network device using any supported protocol, refer to the *Cisco Access Connection Guide*.

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

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 communication server 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

Keepalives 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 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 seconds / 0.2 seconds = 7.5 and 1500 byte packet/ 7.5 = 200 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.	**vty-async mtu** *bytes*

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

TCP running on the device to which the communication server is connected can have a different MTU size than what is configured on the communication server. Because the communication server 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 virtual asynchronous interfaces.

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 communication server.

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

The challenge consists of an ID, a random number, and either the host name of the local communication server 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 communication server 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 communication server.

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 communication server.

CHAP transactions occur only when a link is established. The local communication server does not request a password during the rest of the session. (The local communication server 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 communication server 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 communication server.

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 communication server.

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 Automatic Protocol Startup

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

Task	Command
Configure a line to automatically start an ARA, PPP or SLIP session.	**autoselect** {**arap** | **ppp** | **slip**} | **during login**[1]

1. This command is documented in the "Terminal Line and Modem Support Commands" chapter of the *Access and Communication Servers Command Reference* publication.

The **autoselect** command permits the communication server to allow an appropriate process to start automatically when a starting character is received. The communication server 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 name is presented, 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 to 7. If you change these defaults, the application will not recognize the activation request.

Configure IPX over PPP

You can configure IPX to run over PPP on synchronous serial and asynchronous serial interfaces. To enable IPX over 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 on the communication server.	**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** *network*[1]
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 of this publication.

Configure Performance Parameters

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

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

Compress TCP Packet Headers

You can compress the headers of your TCP/IP packets in order to reduce their size, thereby increasing 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 communication server 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.	**ip tcp header-compression** [**passive**]
Specify the total number of header compression connections that can exist on an interface.	**ip tcp compression-connections** *number*[1]

1. This command is documented in the "IP Commands" chapter of the *Access and Communication Servers Command Reference* publication.

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 the communication server will wait to attempt to establish a TCP connection. In previous versions of communication server 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 communication server) 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 communication server, just to traffic originated at the communication server.

To set the TCP connection attempt time, perform the following task in global configuration mode:

Task	Command
Set the amount of time the communication server will wait to attempt to establish a TCP connection.	**ip tcp synwait-time** *seconds*[1]

1. This command is documented in the "IP Commands" chapter of the *Access and Communication Servers Command Reference* publication.

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 need not be done. Our communication servers generally offer 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-cache1**

1. These commands are documented in the "IP Commands" chapter of the *Access and Communication Servers Command Reference* publication.

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** [*minimum maximum quiet threshold*][1]

1. These commands are documented in the "IP Commands" chapter of the *Access and Communication Servers Command Reference* publication.

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 *Cisco Access Connection Guide* 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 assure 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 seconds / 0.2 seconds = 7.5 and 1500 byte packet/ 7.5 = 200 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** *bytes*

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 communication server is connected can negotiate for a different MTU size than is configured on the communication server. The communication server 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 is such that you might have a high amount of fragments of packets in the output queue.

Improve Asynchronous PPP Performance

To improve asynchronous PPP performance, use the **ppp accm** interface command. In instances where devices have minimal PPP stacks that do not negotiate PPP Asynchronous Control Character Maps (ACCM), the RFC standard default of 0xffffffff is often used, resulting in poor performance.

The **ppp accm** command allows you to set the initial values used by the access or communication server during LCP negotiations with a peer device to alleviate this problem.

Task	Command
Set the value used in negotiation for inbound traffic.	**ppp accm in** *number*
(Optional) Set the value used in negotiation for outbound traffic.	**ppp accm out** *number*
(Optional) Use the same value set for inbound traffic.	**ppp accm match**

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 communication server. Separate access lists can be defined for asynchronous and for other connections.

The tasks described in this section are as follows:

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

Refer to the chapter "Configuring IP," later in this publication, 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 *from* 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 *to* the IP host.	**ip access-group** *access-list-number* **out**

Configure Support for Extended BOOTP Requests

To configure your communication server support to respond to BOOTP requests from client machines, perform the following task in global configuration mode:

Task	Command
Specify the communication server network information that will be sent in response to response to BOOTP requests.	**async-bootp** *tag* [**:***hostname*] *data*

Monitor and Maintain Asynchronous Interfaces

This section describes the following monitoring and maintenance tasks:

- 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** *line-number*
Display parameters that have been set for extended BOOTP requests.	**show async bootp**
Display statistics for asynchronous activity.	**show async status**
Display the status of asynchronous line connections.	**show line** [*line-number*]

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.[1]	**debug ppp chap**

1. Refer to the chapter "Configuring Dial-on-Demand Routing" in this publication 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
- 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
- Remote Network Access Using PPP—Routing IP
- Remote Network Access—Leased Line with Dial-Backup Using PPP

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
```

Note The interface number is the same as the absolute line number, in decimal format. The Cisco 2500 and the 500-CS default to decimal numbers. The ASM-CS displays in octal format. To display line numbers in decimal rather than octal format on the ASM-CS, use the **service decimal-tty** command. Refer to the chapter "System Management Commands" in the *Access and Communication Servers Command Reference* publication for a description of the **service decimal-tty** command.

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 131.108.0.0 0.0.255.255
!
access-list 2 permit 131.108.42.55
access-list 2 permit 131.108.111.1
access-list 2 permit 131.108.55.99
!
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 150.136.79.1
async default ip address 150.136.79.2
ip tcp header-compression passive
```

Conserving Network Addresses Using the IP Unnumbered Feature Example

The following example shows how to configure your communication server for routing using unnumbered interfaces. The source (local) address is shared between Ethernet 0 and async 6 (128.66.1.1). The default remote address is 128.66.1.2.

```
interface ethernet 0
ip address 128.66.1.1 255.255.255.0
!
interface async 6
ip unnumbered ethernet 0
async dynamic routing
```

```
! default address is on the local subnet
async dynamic address
async default ip address 128.66.1.2
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.

```
cs> slip /compressed 1.1.1.1
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 communication server 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 1.0.1.1 255.255.255.0
!
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 1.0.2.1
!
interface async 2
ip unnumbered ethernet 0
async default ip address 1.0.5.1
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 109
network 1.0.0.0
! 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 1.0.0.0
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 communication server 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 1.0.0.2
interface ethernet 0
shutdown
interface async 1
async dynamic routing
ip tcp header-compression on
async default ip address 1.0.5.1
async mode dedicated
ip address 1.0.0.1 255.255.255.0
```

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 communication server is the best place to send most of its packets.

```
router igrp 109
network 1.0.0.0
interface ethernet 0
ip address 1.0.0.1 255.255.255.0
!
interface async 1
async default ip address 1.0.0.101
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 128.148.176.201

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

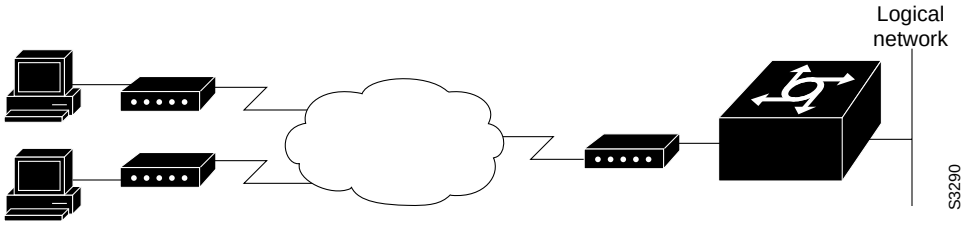
Remote Network Access Using PPP—A Basic Configuration

Figure 15-3 illustrates a simple network configuration comprised of remote PCs with modems connected via modem to a communication server. The cloud is a public switched telephone network (PSTN). The modems are connected via asynchronous lines, and the communication server is connected to a local network.

In this configuration you will need to configure:

- An asynchronous line on the communication server configured to use PPP encapsulation
- An interface on the communication 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 15-3 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
!
int ethernet 0
ip address 182.32.7.1 255.255.255.0
!
interface async 1
encapsulation ppp
async mode interactive
async default ip address 182.32.7.51
async dynamic address
ip unnumbered ethernet 0

line 1
autoselect ppp
modem callin
speed 19200
```

Remote Network Access Using PPP—Routing IP

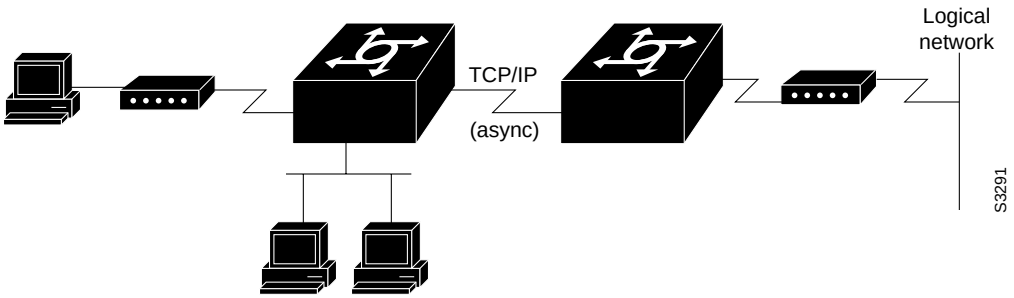
Figure 15-4 illustrates a network configuration that provides routing functionality, allowing routing updates to be passed across the asynchronous lines.

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

For this scenario, you will need to configure:

- An asynchronous line on both communication servers configured to use PPP encapsulation
- An interface on both communication 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 15-4 Routing On an Asynchronous Line Using PPP



The configuration is as follows:

```
interface async 1
encapsulation ppp
async mode interactive
async default ip address 182.32.7.51
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 communication servers, starting in global configuration mode:

```
interface async 2
async default ip address 182.32.7.55
ip tcp header compression passive
```

Finally, configure routing as described in the *Router Products Configuration Guide*, 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).

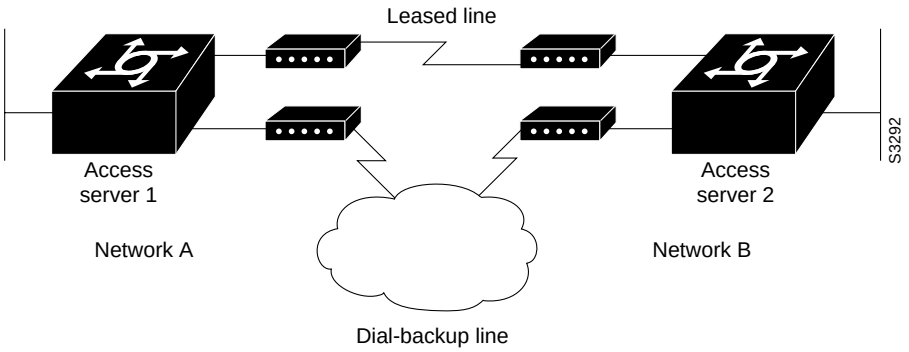
Remote Network Access—Leased Line with Dial-Backup Using PPP

Figure 15-5 illustrates a scenario where two networks are connected via communication 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:

- Two asynchronous interfaces on each communication server
- Two modem interfaces
- A default IP address for each interface
- Dial-backup on one modem interface per communication server
- An interface connecting to the communication server's related network

Figure 15-5 Figure 14-3 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 199.199.199.1 255.255.255.0
!
interface Async1
ip address 131.108.199.1 255.255.255.0
encapsulation ppp
async default ip address 131.108.199.2
async dynamic address
async dynamic routing
async mode dedicated
dialer in-band
dialer map IP 131.108.199.2 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
```