Chapter 3
Protocol Translator User Commands 3

This chapter describes how to use the Cisco protocol translator. You will find information about these tasks in this chapter:

- Making X.3 PAD connections.
- Starting protocol translation sessions.
- Making connections to the supported protocols—DEC LAT, Telnet, UNIX rlogin, TN3270, XRemote, and X.25 terminal support.
- Managing connections and sessions.
- Locally changing the terminal settings or *parameters*.
- Accessing Cisco's implementation of the Digital Equipment Corporation (DEC) Maintenance Operation Protocol (MOP) server.

Refer to the chapter "Startup and Basic Configuration" for configuration procedures to use the **setup** command facility. Refer to the chapter "System Configuration" for details about system-wide configuration features. The chapter "System Management" addresses facilities for monitoring and managing Cisco protocol translators. If you have special applications or wish to customize your connections, refer to the specific transmission protocol chapters for details.

See the end of this chapter for a command summary.

X.3 PAD Connections

A PAD is a packet assembler/disassembler, which is a device that collects data from a group of terminals and periodically outputs the data in *packets* (data organized in a special format). A PAD also does the reverse. It can take data packets from a host and return them into a character stream that can be transmitted to the terminals. A PAD is defined by CCITT Recommendations X.3, X.28, and X.29.

For a discussion and sample scenarios of one-step and two-step connections using PAD and Telnet, refer to the section "Protocol Translation Methods" in this chapter.

CCITT Recommendation X.3 specifies the parameters for terminal-handling functions such as baud rate, flow control, character echoing, and so on, for a connection to an X.25 host. The X.3 parameters are similar in function to the Telnet options.

CCITT Recommendation X.29 specifies a protocol for setting the X.3 parameters via a network connection. When a connection is established, the destination host can request that the PAD or terminal change its parameters using the X.29 protocol. A PAD can refuse to do this, in which case a terminal user can change the parameter later. A PAD cannot tell the destination host to change its X.3 parameters, but it can communicate that its own parameters were changed.

Along with Recommendations X.3 and X.29, the CCITT also provides Recommendation X.28 to specify the user interface for locally controlling a PAD; however, the protocol translator is not a PAD and this recommendation is not supported.

Making PAD Connections

You use the **pad** EXEC command to initiate outgoing connections to a PAD host. Enter this command at the EXEC prompt:

pad *x121-address* **[/cud** *text***] [/debug] [/reverse]**

The argument *X121-address* is the X.121 address of the X.25 host. If the host-to-address mapping has been set with the **x25 host** command, you can use the host name instead of the address. (The **pad** command supports one-word connections; in other words, you are not required to enter the **pad** command, just the address is enough to start connection.)

The optional keyword and argument **/cud** *text* is a switch that includes *text* in the Call User Data field of the outgoing Call Request Packet.

The optional keyword **/debug** is a switch that causes the informational level of logging messages to be printed whenever the remote host changes an X.3 parameter setting or sends any other X.29 control packet.

The optional keyword **/reverse** is a switch that causes Reverse Charge calls to be accepted on a per-call, rather than a per-interface, basis.

Example:

To connect to PAD host address *77630*, enter this command:

pad 77630

The X.3 parameters are normally set by the host system; however, there are occasions when the PAD is connected as a front end for some other computer or device that is not configured to set these parameters. To accommodate such systems, an X.29 READ PARAMETER message is sent immediately after opening the connection.

If the connection is made to a PAD, a PARAMETER INDICATION message is returned, and the X.3 parameters are set to match those required by the PAD. If the connection is made to a host, the READ request is ignored and an X.29 SET PARAMETER message is sent. If a host sends both a PARAMETER INDICATION and X.29 SET PARAMETER message, the X.29 SET PARAMETER message takes precedence.

You use the standard escape sequence to return to the EXEC. Multiple outgoing PAD connections are supported.

You use the **resume** command to resume a connection. The **resume** command also has a feature for setting X.3 PAD parameters, as explained in the subsection “Setting X.3 PAD Parameters” in this chapter.

Handling Multiple PAD Connections

You can have any number of PAD connections at a time and you can switch back and forth among them. To switch from one connection to another, first type the escape sequence (Ctrl ^ X by default) to return to the EXEC. Then use the **pad** command to make the new connection.

Use the EXEC **where** command to check the logical name or number of a connection. Use the EXEC **resume** command to return to a previous connection.

Using the PAD Resume Command

Once you have the connection number from the **where** command, use the **resume** command and the connection number to resume the connection. The command syntax is:

resume [*connection***] [/***keyword***] [/set** *parameter:value***]**

This command also has a feature for setting X.3 PAD parameters, as explained in “Setting X.3 PAD Parameters.” Local X.3 parameters can be changed during a connection by escaping back to the EXEC and issuing the **resume** command with the appropriate parameter set.

The optional argument *connection* is a connection name or number; the default is the most recently used connection.

The optional */keyword* specifies line options for the connection, and can be one of the following:

- **/line**—Enables line-mode editing.
- **/noline**—Disables line mode and enables character-at-a-time mode, which is the default.
- **/debug**—Displays informational messages whenever the remote host changes an X.3 parameter or sends an X.29 control packet.
- **/nodebug**—Disables the informational display, which is the default.
- **/echo**—Enables local echo, which is the default.
- **/noecho**—Disables local echo.

The **/set** switch is described in the next section.

Setting X.3 PAD Parameters

You can change local X.3 parameters during a connection by using either the **/set** switch of the **resume** command or the **x3** EXEC command.

Example:

To reset the outgoing connection default for local echo mode, enter this command:

```
resume 3 /set 2:1
```

The **/set** switch sets the X.3 parameters defined by parameter number and value, separated by a colon. These parameters and their values are described in the "X.3 PAD Parameters" section. For outgoing connections, the X.3 parameters default to the following:

```
2:1, 3:2, 4:1, 7:4, 16:127, 17:21, 18:19
```

All other parameters default to zero, but can be changed using the **/set** switch.

For incoming PAD connections, software sends an X.29 SET PARAMETER packet to set only the following parameters:

```
2:0, 4:1, 7:21, 15:0
```

If you remain at the EXEC prompt, use the **x3** command to set local PAD parameters. Parameters are set directly, without using a command switch.

Example:

This example resets the outgoing connection default for local echo mode.

```
x3 2:1
```

The X.3 PAD Parameters

Following are descriptions of the X.3 parameters. Default values are noted in the descriptions. The default value for any parameter not so noted is zero for outgoing connections or not set for incoming PAD connections. For incoming PAD connections, the protocol translator sends an X.29 SET PARAMETER packet to set the noted defaults.

Since the X.3 parameters describe the user's terminal, which exists on only one side of the connection, the PAD protocols are not always symmetric.

Some of these commands require ASCII decimal values which are listed in the appendix "ASCII Character Set."

Parameter 1: Escape From Data Transfer (Not Supported)

Parameter 1 determines whether or not the protocol translator will be allowed to escape from data transfer mode in order to send PAD command signals. Since the Cisco EXEC uses a two-character escape sequence, and there is no way to set the escape character on a Telnet connection, this parameter is refused on translation sessions as well.

Parameter 2: Local Echo Mode

Parameter 2 determines whether or not PAD is required to perform local echo of characters. This parameter can be negotiated end-to-end on translation sessions. On incoming PAD connections, software turns local echo off on the remote PAD to support the Cisco user interface.

Table 3-1 PAD Local Echo Mode Values

Value	Description
0	No local echo (incoming PAD connection default).
1	Local echo on (outgoing connection default).

Parameter 3: Data Forward Character

Parameter 3 sets up a packet forwarding mask, that is, it selects which character causes PAD to forward a packet either before expiration of the Idle Timer (see Parameter 4) or when in local editing mode.

Table 3-2 PAD Data Forward Characters Values

Value	Description
0	None - full packet.
1	Forward packet upon receipt of an alphanumeric character.
2	Forward packet upon receipt of a RETURN (outgoing connection default).
4	Forward packet upon receipt of ESCAPE, BEL, ENQ, or ACK.
8	Forward packet upon receipt of DEL, CAN, OR DC2.
16	Forward packet upon receipt of ETX or EOT.
32	Forward packet upon receipt of HT, LT, VT, or FF.
64	All other characters in the ASCII chart.

As X.3 supports a wider variety of dispatch characters than Telnet does, parameter changes to or from the default causes a translation session to negotiate in or out of line mode on the Telnet connection.

A forwarding mask can be statically set using the Cisco **terminal dispatch-character** terminal parameter-setting EXEC command. This command can set any character or characters as the forwarding mask, and overrides (when logical) any values set by parameter 3.

Parameter 4: Idle Timer

Parameter 4 controls the amount of time the software waits for new data before sending a packet in the absence of a data forwarding character.

Table 3-3 PAD Idle Timer Values

Value	Description
0	No timer.
1	Do not delay before sending a packet in the absence of a data forwarding character. This is the default.
2-255	Delay time before sending a packet, in twentieths of a second.

Parameter 5: Device Control (Not Supported)

Parameter 5 selects whether PAD can transmit flow control (ASCII XON/XOFF) characters during data transfer to the terminal to control the terminal and data flow. Flow control is not directly supported on protocol translators because data must make network hops to travel to its final destination. However, depending on the type of incoming connection, setting this parameter can cause similar negotiations to be sent over the connection, thereby attempting to change the state of the flow control option at the device closest to the user.

Parameter 6: PAD Service Signals (Not Supported)

Parameter 6 selects whether or not PAD is required to transmit service signals. As the protocol translator does not use Recommendation X.28 for its user interface, this parameter is ignored.

Parameter 7: Action Upon Receipt of a BREAK Signal

Parameter 7 defines the action of the PAD after receiving a BREAK signal from the terminal.

Table 3-4 PAD BREAK Signal Values

Value	Description
0	Ignore the BREAK signal.
1	Transmit an interrupt packet to notify the remote host or another PAD that the BREAK signal has been generated.
2	Transmit a Reset packet to reset the virtual circuit.
4	Transmit an X.29 Break indication to the remote host, or to a PAD (out-going connection default).
8	Escape from data transfer mode.
16	Discard output to the terminal by setting parameter 8 to a value of 1.
21	Combination of values 1, 4 and 16 (incoming connection default).

The PAD protocols allow you to send a special Indication of Break X.29 packet, send an Interrupt packet, perform a Reset operation, act as if the Recall character had been typed, or begin discarding output to the user. Combinations of these options are also allowed, as long as they make sense. Common options are to begin discarding output and send both an X.25 Interrupt packet and an X.29 Indication of Break packet, and these options are supported. All other options are not supported and are silently ignored.

Parameter 8: Discard Output

Parameter 8 indicates to the PAD whether to discard received packets rather than disassemble and transmit them. This parameter works in conjunction with parameter 7. If value 16 is chosen for parameter 7, all output is discarded after reception of the BREAK signal. Setting parameter 8 to zero restores normal data delivery to the terminal.

Table 3-5 PAD Discard Output Values

Value	Description
0	Normal data delivery to the terminal (outgoing connection default.)
1	Discard all output to the terminal. Set by parameter 7; see previous description.

This parameter can also be set and unset manually using the PAD **resume** EXEC command.

Parameter 9: Return Padding (Not Supported)

Parameter 9 determines whether or not PAD should provide padding (insert filler characters) upon receipt of a Return character from the terminal.

Parameter 10: Line Folding (Not Supported)

Line folding means inserting a LINE FEED at a certain point which places subsequent characters on the next line. Parameter 10 determines selection of this function and specification of the line length.

Parameter 11: Baud Rate

Parameter 11 is a read-only value that determines the baud rate transmitted across the interface between PAD and the terminal.

Table 3-6 PAD Baud Rate Values

Value	Description (in bits per second)
10	50
5	75
9	100
0	110
1	134.5
6	150
8	200
2	300
4	600
3	1200
7	1800
11	75/1200
12	2400
13	4800
14	9600
15	19200
16	48000
17	56000
18	64000

Parameter 12: Input Flow Control (Not Supported)

Parameter 12 determines whether or not the terminal can transmit ASCII XON/XOFF (transmission on and off) characters to PAD during the data transfer mode. Flow control is not directly supported on protocol translators because data must make network hops to travel to its final destination. However, depending on the type of incoming connection, setting this parameter can cause similar negotiations to be sent over the connection, thereby attempting to change the state of the flow control option at the device closest to the user.

Parameter 13: LINE FEED Insertion

Parameter 13 determines the procedure for inserting the LINE FEED character upon receipt of a RETURN character. The PAD also responds to a value that results from the addition of any of the following values inTable 3-7.

Table 3-7 PAD LINE FEED Signal Values

Value	Description
0	Do not insert the LINE FEED signal (outgoing connection default).
1	Insert a LINE FEED after transmitting RETURN to the terminal.
2	Insert a LINE FEED after echoing RETURN to the terminal.
4	Insert a LINE FEED after echoing RETURN to the remote host.

Parameter 14: LINE FEED Padding (Not Supported)

Parameter 14 determines whether or not PAD should provide padding (insert filler characters) upon receipt of a LINE FEED character from the terminal. This function is generally provided by the end user's operating system.

Parameter 15: Local Editing

Parameter 15 enables or disables a PAD editing function for the terminal in data transfer mode. Enabling the editing function disables the Idle Timer (see Parameter 4). The user at the terminal can make corrections and display the line buffer containing the characters to be transmitted when the forwarding character (see parameter 3) is received.

Table 3-8 PAD Local Editing Function

Value	Description
0	Disables editing capabilities in data transfer mode. Any characters entered become part of the data stream and are transmitted (default for both connection types).
1	Enables editing capabilities in the data transfer mode which suspends the following PAD operations: 1) Full packet data forwarding until the edit buffer is full, and 2) Forwarding of DATA packets upon expiration of the Idle Timer.

Parameters 16, 17, and 18 provide the editing functions.

Parameter 16: Character Delete

Parameter 16 allows you to select a character that will delete characters while in PAD editing mode. This character is valid only if parameter 15 is set to one.

Table 3-9 PAD Character Delete Editing Function

Value	Description
0-127	Select one character from the ASCII character set to represent the delete character. Default is character 127 (DEL).

Parameter 17: Line Delete

Parameter 17 allows you to select a character that will delete a line while in PAD editing mode. This character is valid only if Parameter 15 is set to one.

Table 3-10 PAD Line Delete Editing Function

Value	Description
0-127	Select one character from the ASCII character set to represent the delete character. Default is character 21 (CTRL-U).

Parameter 18: Line Display

Parameter 18 allows you to select a character that will display a line while in PAD editing mode. This character is valid only if Parameter 15 is set to one.

Table 3-11 PAD Line Display Editing Function

Value	Description
0-127	Select one character from the ASCII character set to represent the delete character. Default is character 18 (CTRL-R).

Protocol Translation Methods

Protocol translation allows two systems running different networking protocols to communicate as if the protocols were the same. This can be a difficult task, as networking protocols may require different numbers of network connections to run an application, or may have different formats for their data and commands.

For this reason, there may never be a protocol translator that handles electronic mail or file transfer in a completely transparent manner. Fortunately, there are fewer differences among protocols that connect remote terminals to a computer (called virtual terminal protocols).

The protocol translator attempts to provide transparent protocol translation between systems running disparate protocols. Cisco protocol translation fully supports two-way virtual terminal protocol translation between nodes running X.25, DEC LAT, and TCP/Telnet. Limited translation (two-step only) is supported for XRemote (to X.25 PAD environments) and TN3270 (to LAT, X.25, and TCP/Telnet). This allows terminal users on one network to access hosts (or host-like devices) on another network, despite differences in the native protocol stacks associated with the originating device and the target host. You can use either of two methods to accomplish this, as discussed in the following sections.

Making One-Step Connections

Note: One-step translation applies only to connections involving TCP (Telnet), LAT, and X.25. Protocol translators do not support one-step translations for TN3270 or XRemote.

The one-step method invokes a *translation session* process to provide fully transparent protocol conversion. In this method, the protocol translator masquerades as two or more hosts on the same network. When a connection is made to the protocol translator, it determines which host the connection is for, and what protocol that host is using. It then establishes a new network connection, using the networking protocol required by that host.

This connection is more efficient, and allows the protocol translator to act upon greater knowledge of the protocols in use, as the protocol translator acts as a network connection rather than a terminal. One disadvantage, however, is that the initiating computer or user does not know that two networking protocols are being used. This creates the limitation that parameters of the foreign network protocols cannot be changed, once the connections are established. The exception to this limitation is the set of parameters common to both networking protocols. Any parameter in this set can be changed from the first host to the final destination.

To support connections in each direction, you must specifically include **translate** command statements in the configuration file to handle bidirectional connections.

Note: Refer to the chapter “Protocol Translation” for more details concerning how to configure protocol translators for one-step translation using the **translate** command and for examples of how to build one-step connection environments.

Using the One-Step Method to Connect (TCP to X.25 Host)

A typical one-step translation application might feature a UNIX workstation user attempting a connection to a remote X.25 host, named *host1*, over an X.25 PDN. In this case, the protocol translator automatically converts the Telnet connection request to an X.25 connection request and transmits the request as specified in the system configuration.

A connection is established by first entering the **telnet** command at the UNIX workstation system prompt, as follows:

```
unix%telnet host1
```

Note: This example implicitly assumes that the name *host1* is known to the UNIX host (obtained via DNS, IEN116, or a static table) and is mapped to the IP address used in a **translate** command. Refer to the chapter "Protocol Translation" for examples of how to define and use the **translate** command.

The protocol translator accepts the Telnet connection, and immediately forms an outgoing connection with the remotely located *host1* as defined in a **translate** command in your protocol translator's active configuration file. Next, *host1* sets several X.3 parameters, including local echo. Since the Telnet connection is already set to local echo (at the UNIX host), no changes are made on the TCP connection.

The *host1* connection prompts for a user name, which is provided (and echoed locally), as follows:

```
login name:
```

Then *host1* sets the X.3 parameter to cause remote echo (the same process as setting X.3 PAD parameter 2:0), and prompts for a password. The protocol translator converts this to a Telnet option request on the UNIX host, which then stops the local echo mode.

At this point the user is connected to the PAD application and the application will set the X.3 PAD parameters (although they can always be overridden by the user). When the user is finished with the connection, he enters the escape character to exit back to the host connection, then enters the appropriate command to close the connection.

The *host1* host immediately closes the X.25 connection. The protocol translator then drops the TCP connection, leaving the user back at the UNIX system prompt.

Making a Two-Step Connection

In the two-step method, you explicitly establish a network connection to a protocol translator, and from there establish an outgoing network connection on a different type of network. The two steps are:

Step 1: Establish the incoming connection directly to the protocol translator.

Step 2: Establish the outgoing connection from the protocol translator to another network host.

Note: In general, the two-step process is applicable when you want to use a protocol translator as a *general purpose gateway* between two types of networks (for example, X.25 PDN and TCP/IP). Instead of configuring every possible connection via embedded **translate** commands, the two-step method allows you greater flexibility in terms of connecting to network resources accessible via the protocol translator. The two-step process also allows another level of security if TACACS global commands or the **login** line subcommand are configured. These security features are described in the chapter "System Configuration."

This process is similar to connecting a group of terminal lines from a PAD, to a group of terminal lines from a TCP protocol translator. The difference is that you do not encounter the wiring complexity, unreliability, management problems, and performance bottlenecks that occur when two separate devices are connected via asynchronous serial lines. With the two-step connection process, you can individually modify the parameters of either network connection, even while a session is in process.

The protocol translator supports the two-step model in both directions (for example, from Telnet to PAD, and the reverse).

Note: You must use the two-step method for translations of TN3270 and XRemote. Refer to the "TN3270 Configuration and Management" and "XRemote Configuration and Management" chapters for more information concerning these protocols.

Changing Parameters and Settings Dynamically

If you use the two-step method to connect, you can change PAD and local terminal parameters dynamically during a session.

Sample Session:

The following sample session shows how to make a dynamic change during a session. In this example, you want to type in some information at a remote host called the *Information Place.* Suppose that you need to change the X.3 PAD parameters that define the editing characters from the default Delete key setting to the Ctrl-D sequence.

To do this, first enter the escape sequence to return to the system EXEC:

```
Ctrl ^ x
```

Then at the EXEC, type in the **resume** command with the **/set** keyword and the desired X.3 parameters.

```
pt>resume /set 16:4
```

You also want to set the **/debug** switch to check that your parameter setting has not been changed by the host PAD. The **/debug** switch provides helpful information about the connection.

At the EXEC, you can also set a packet dispatch character or sequence using the **terminal dispatch-character** command. The **dispatch-character** line subcommand is discussed in the chapter “System Configuration.” This example shows how to set ESC as a dispatch character using the EXEC command version of **dispatch-character**:

```
pt>terminal dispatch-character 27
```

To return to the PAD connection, simply enter:

```
pt>resume
```

Now you can use your new settings for your session.

Connecting Using the Two-Step Method (TCP to PAD)

The first step in any two-step translation is to connect directly from a terminal or workstation to a protocol translator. For example, you might make the following connection request:

```
unix_host%telnet pt
```

If a protocol translator named *pt* can be reduced, it returns a login message and you enter your login name and password.

The next step then is to connect from the protocol translator to the target system. For example, you connect to an X.25 host using the **pad** command by entering the following:

```
pt>pad 71330
```

Once the connection is established, the protocol translator immediately sets the PAD to single character mode with local echoing, since this is the behavior the protocol translator expects. The PAD responds with its login messages and a prompt for a password:

```
Trying 71330...Open
Welcome to the Information Place
Password:
```

As the password should not echo on your terminal, the PAD requests remote echoing so that characters will be exchanged between the PAD and the protocol translator, but not echoed locally or displayed. After the password is verified, the PAD again requests local echoing from the protocol translator, which it does from then on.

To complete this sample session, you log off, which returns you to the protocol translator system EXEC prompt. From there, you execute the **quit** command and the protocol translator drops the network connection to the PAD.

Connection Capabilities

The Cisco protocol translator provides a broad range of connection service capabilities. Users can make simple automated terminal-to-host connections, or can establish custom terminal sessions tailored to specific terminal or application requirements. The connection descriptions in this section define the connection capabilities available to users and the syntax associated with each user level EXEC connection command.

Making DEC LAT Connections

DEC's Local Area Transport (LAT) protocol is the protocol used most often to connect protocol translators to DEC hosts. LAT is a DEC-proprietary protocol. Cisco uses LAT technology licensed from DEC.

The LAT protocol allows a user at one site to establish a connection to a host at another site, then passes the keystrokes from one system to the other. A user can establish a LAT connection through the protocol translator to a DEC host, simply by entering the host name.

Use the EXEC command **lat** to establish a connection to a LAT learned service. Enter this command at the EXEC prompt:

lat *name* **[node** *nodename* **| port** *portname* **| /debug]**

You can type the name of the service to which you want to attach, or include the optional keywords listed in the above syntax description. You can also just type the service name without the **lat** command to make the connection.

The optional keyword **node** specifies connection to a specific LAT node which offers a service. Enter the name of the node in place of the *nodename* argument. If you do not include the node option, the node with the highest rating offering the service is used. Use the EXEC command **show lat nodes** to display information about all known LAT nodes (see the section "Monitoring LAT" in the chapter "LAT Configuration and Management" for more information about this command).

The optional keyword **port** specifies a destination LAT port name. This keyword is ignored in most timesharing systems, but is used by protocol translators offering *reverse LAT* services. *Reverse LAT* services are discussed in the section "Inbound Session Support," in the chapter "LAT Configuration and Management." Generally, *reverse LAT* involves connecting to one communications server (protocol translator or terminal server) from another. In this case, the target server runs the host portion of the protocol. Enter the port name in the format of the remote system in place of the *portname* argument.

The optional keyword **/debug** is a switch that, when enabled, prints parameter changes and other special messages on the terminal.

Note: With Cisco's implementation of LAT, you are not required to enter the word **lat** to establish a LAT connection. If you prefer, you can just enter the LAT learned service name. To define a preferred transport protocol for a specific line, refer to the section "Defining the Transport Protocol for a Specific Line" in the chapter "System Configuration." To show a listing of the available LAT learned services, use the **show lat services** command, described in the section "Monitoring LAT" in the chapter "LAT Configuration and Management."

Sample Session 1:

The following sample session establishes a LAT connection from the protocol translator named *pt* to the host *Eng2*.

```
pt>lat eng2
Trying ENG2...Open
         ENG2 – VAX/VMS V5.2
Username:JSmith
Password:
    Welcome to VAX/VMS version V5.2 on node ENG2
    Last interactive login on Friday,  6-APR-1990 19:46
```

The LAT protocol is explicitly specified. You specify the protocol when your preferred transport is set to "none" or to another protocol. The system informs you of the sequence of connection events by displaying "Trying <system>..." and then "Open." If the connection were not successful, you receive a failure message.

Sample Session 2:

The following sample session establishes a LAT connection from the protocol translator named *pt* to the *cisco-modems* service and specifies *port 24*, which is a special modem:

```
pt>lat cisco-modems port 24
```

Sample Session 3:

The following sample session establishes a LAT connection from the protocol translator named *pt* to the *cisco-modems* service and specifies a node named *Eng*:

```
pt>lat cisco-modems node eng
```

Sample Session 4:

The following sample session uses the LAT session debugging capability.

```
pt>lat Eng2 /debug
Trying ENG2...Open
        ENG2 – VAX/VMS V5.2
```

```
    Username:**JSmith**
    Password:
       Welcome to VAX/VMS version V5.2 on node ENG2
       Last interactive login on Tuesday, 10-APR-1990 19:02
   [Set Flow out off, Flow in on, Format 8:none, Speed 9600/9600]
   [Set Flow out off, Flow in on, Format 8:none, Speed 9600/9600]
   $ set ter/speed=2400
   [Set Flow out off, Flow in on, Format 8:none, Speed 2400/2400]
```

A variety of LAT events are reported, including all requests by the remote system to set local line parameters. The messages within brackets ([ ]) above are the messages produced by the remote system setting line characteristics to operating system defaults.

You may open more than one connection by typing the escape sequence Ctrl^X (done by typing and holding the Ctrl, SHIFT and ^ keys, then releasing those keys and typing the X key) to return to the EXEC prompt. Enter the **show sessions** EXEC command to display open sessions. Enter the **resume** command with the connection number desired to return to a connection. (See the section "Connection Management Commands" in this chapter for further information about these command.)

Making LAT Connections to Password Protected Services

When you attempt to connect to a password-protected service, the protocol translator prompts for a password. You can abort the connection by typing Ctrl-C, or enter the password for a given service to complete the connection.

Making and Managing Telnet Connections

Two TCP/IP protocols, Telnet and rlogin, are available for making connections to a host. Telnet, a virtual terminal protocol that is part of the TCP/IP protocol suite, allows for connections to hosts. Telnet is the more widely used protocol. The rlogin protocol is a remote login service developed for the BSD UNIX system. It provides better control and output suppression than Telnet, but may only be used when the host (typically, a UNIX system) also supports rlogin.

To make a Telnet connection, type the name or address of the host to which you want to connect at the EXEC prompt and press Return. Type the **connect** command before the host name, if the name of the host is the same as a command word. To make an rlogin connection, type the **rlogin** command before the host name or address. In either case, the protocol translator either connects you to the host or prints a message stating that the connection failed.

When you are done, log off the host. The EXEC terminates any connection after ten minutes (default) of inactivity, or when you enter the **exit** command at the EXEC prompt.

However, the Cisco software also provides additional connection options. The following sections explain these options, and commands to display and monitor the connections.

Making Telnet Connections

Use either the EXEC command **connect** or **telnet** to start a Telnet connection; the commands are synonymous. Enter this command at the EXEC prompt.

> **[connect|telnet]** *host* **[***port***]** **[***/keyword***]**

The argument *host* is a host name or an Internet address. The optional argument *port* is a decimal TCP port number; the default is the Telnet server port (decimal 23) on the host. The optional argument *keyword* is one of the following:

- **/route** *path*—Specifies loose source routing. The argument *path* is a list of host names or Internet addresses that specify network nodes, ending with the final destination.
- **/line**—Enables Telnet line mode. In this mode, the protocol translator does not send any data to the host until you press Return. You can edit the line using the standard protocol translator command editing characters (Backspace, Delete, Ctrl-U, Ctrl-W). The **/line** keyword is a local switch; the remote server is not notified of the mode change.
- **/debug**—Enables Telnet debugging mode.
- **/stream**—Turns on *stream* processing, which enables a raw TCP stream with no Telnet control sequences. A stream connection does not process Telnet options, and may be appropriate for connections to ports running UUCP and other non-Telnet protocols.

Note: With Cisco's implementation of TCP/IP, you are not required to enter the word **connect** or **telnet** to establish a Telnet connection. If you prefer, you can just enter a host name. To show a listing of the available hosts, use the **show hosts** command (see the section "Monitoring TCP/IP" in the chapter "TCP/IP Configuration and Management" for more information about this command).

You may omit the command word **connect** or **telnet**, if the host name you want to use is not the same as a protocol translator command word.

The protocol translator assigns a logical name to each connection; several commands use these names to identify connections. The logical name is the same as the host name, unless that name is already in use or you change the connection name with the EXEC command **name-connection**. If the name is already in use, the protocol translator assigns a null name to the connection.

Examples:

The following command routes packets from the source system to *kl.sri.com*, then to 10.1.0.11, and finally to *mathom*.

```
PT>connect mathom /route:kl.sri.com 10.1.0.11 mathom
```

The following command connects to a host with logical name *mathom*.

```
PT>mathom
```

Executing Special Telnet Commands

The Cisco Telnet software supports special Telnet commands in the form of Telnet sequences that map generic terminal control functions to operating system-specific functions.

To issue a special Telnet command, type the escape sequence (usually Ctrl ^) and then a command character. You can type the command character as you hold down Ctrl or with Ctrl released; you can type either uppercase or lowercase letters.

Table 3-12 lists the special Telnet commands.

Table 3-12 Special Telnet Commands

Command Name	Command Character
Break	Ctrl-^,B
Interrupt Process (IP)	Ctrl-^,C
Erase Character (EC)	Ctrl-^,H
Abort Output (AO)	Ctrl-^,O
Are You There? (AYT)	Ctrl-^,T
Erase Line (El)	Ctrl-^,U

At any time during an active Telnet session, you can list the Telnet commands by typing this command at the system prompt:

Ctrl ^ ?

This is done by typing the escape sequence followed by a question mark. This command displays an online table of the special Telnet commands, for quick reference.

A sample of this list follows (the Ctrl key is represented by the first ^ character).

```
[Special telnet escape help]
^^B  sends telnet BREAK
^^C  sends telnet IP
^^H  sends telnet EC
^^O  sends telnet AO
^^T  sends telnet AYT
^^U  sends telnet EL
```

Making Rlogin Connection

The **rlogin** command starts a connection using the UNIX rlogin protocol. Enter this command at the EXEC prompt.

rlogin *host* **[debug]**

The argument *host* is a host name or an Internet address. The optional keyword **debug** enables debugging output from the rlogin protocol.

The rlogin protocol provides better flow control and output suppression than Telnet. The Cisco implementation of rlogin does not subscribe to the trusted host model. That is, a user cannot automatically log onto a UNIX system from the protocol translator, but must provide a user ID and a password for each connection.

Example:

This example makes an rlogin connection to a host at address *108.33.21.2* and enables the message mode for debugging.

```
PT>rlogin 108.33.21.2 debug
```

Establishing Multiple Telnet Connections

You can have several concurrent Telnet or rlogin connections open and switch back and forth between them. The number of connections that may be opened is defined by the **session-limit** command; see the chapter “System Configuration” for more information about this command. To switch between connections, first type the escape sequence, which by default is Ctrl ^ X (press the Ctrl, Shift and ^ keys simultaneously, let go, then press the X key), to return to the system command prompt.

To make a new connection, use the procedure described in the sections “Making Telnet Connections,” or “Making Rlogin Connections,” above. To get back to an existing connection, use the **resume** command, described next. Use the **where** command to list your open connections.

Switching Between Telnet Connections

The EXEC command **resume** returns to a previous connection. Enter this command at the EXEC prompt:

resume [*connection***] [***/keyword***]**

The optional argument *connection* is the connection name or number; the default is the most recent connection. The optional argument */keyword* is one of the following keywords:

- **/line**—Enables Telnet line mode. In this mode, the protocol translator does not send any data to the host until the user presses Return. The user can edit the line using the standard protocol translator command editing characters (Backspace, Delete, Ctrl-U, Ctrl W). The **/line** keyword is a local switch; the remote server is not notified of the mode change.
- **/noline**—Disables Telnet line mode and enables character-at-a-time mode (default).
- **/debug**—Enables Telnet debugging mode.
- **/nodebug**—Disables Telnet debugging mode (default).
- **/stream**—Turns on stream processing, which enables a raw TCP stream with no Telnet control sequences. A stream connection does no processing of Telnet options, and may be appropriate for connections to ports running UUCP and other non-Telnet protocols.

- **/nostream**—Turns off stream processing, which enables the Telnet protocol (default).
- **/echo**—Enables local echoing of characters (default). The **/echo** keyword is a local switch; the remote server is not notified of the state change.
- **/noecho**—Disables local echoing of characters.

Note: The protocol translator supports Telnet line mode, also called local editing; you can request line mode with the **connect** command. If a remote host responds with "WONT SUPPRESS-GA," the protocol translator assumes the host wants line-at-a-time input along with line mode. You can also put a Telnet session into line mode by using the **/line** keyword with the **resume** command.

You can omit the command word **resume** and simply type the connection number to resume a connection. You can also return to the most recent session by just pressing the Return key.

Examples:

This command resumes connection 2 in Telnet line mode.

```
PT>resume 2 /line
```

This command resumes Telnet connection 3.

```
PT>3
```

Listing Open Telnet Connections

The EXEC command **where** displays information about all open Telnet or rlogin connections associated with the current terminal line. Enter this command at the EXEC prompt:

where

Sample Session:

This session shows the display seen by entering the **where** EXEC command.

```
PT>where

Conn Host                Address            Byte  Idle Conn Name
   1 DREGGS           131.108.19.50    0     0   DREGGS
*  2 CLASH            192.31.7.24      0     0   CLASH
```

The information it displays includes the host name, address, number of characters waiting to be sent to the terminal, idle time, and connection name. An asterisk (*) indicates the current connection.

Making TN3270 Connections

If your network administrator has configured an appropriate default terminal type, then you can use the EXEC command **tn3270** to start a TN3270 session via a Cisco protocol translator. Enter this command at the EXEC prompt:

tn3270 *hostname*

The argument *hostname* is the name of a specific host on a network that is reachable by the protocol translator. The default terminal emulation mode allows access using a VT100 emulation. If your terminal requires a different emulation, you must configure your protocol translator to support different terminal types.

Note: Unlike Telnet and LAT connections, you *must* enter the command **tn3270** in order to make a connection to a host with this command.

The process for configuring the Cisco protocol translator to include alternative (and custom) terminal emulations is described in the chapter "TN3270 Configuration and Management."

Example:

The following example allows the Cisco protocol translator to establish a terminal session with an IBM host named *finance*.

```
pt>tn3270 finance
```

Making XRemote Connections

XRemote can be invoked in these ways:

- Automatically using the XDMCP (X Display Manager Control Protocol) for TCP/IP networks
- Automatically using DECwindows login for LAT networks
- Manually via a step-by-step access process

The following brief procedures outline steps required for starting up XRemote in several typical environments. When possible, use the automated processes. See the chapter "XRemote Configuration and Management" for information about configuring XRemote and fonts for screen display.

Automatic Session Startup—XDMCP Server

If your host computer supports a server for XDMCP, such as the *xdm* program included in X11R4 or later, you can use automatic session startup to make an XRemote session connection. Use the EXEC command **xremote xdm** to initiate XRemote using XDMCP.

xremote xdm *hostname*

This command causes an XDMCP session start up request to be made to the computer specified (*hostname*). If a host name is not specified, a broadcast message is sent to all hosts. The first host to respond by starting up a session is used.

The protocol translator and X terminal stay in XRemote mode until either the display manager terminates the session, or a reset request is received from the X terminal.

Example:

This command starts a session with a remote host named VMS-1.

```
PT>xremote xdm VMS-1
```

Automatic Session Startup—DECwindows Login via LAT

If your host computer supports DECwindows login sessions, you can use automatic session startup to make an XRemote session connection. Once the system administrator at the remote host has configured support for DECwindows over LAT, you can use the EXEC command **xremote lat** to initiate the connection. Enter this command at the EXEC prompt:

xremote lat *service*

The argument *service* is the name of the desired LAT service.

When you issue this command, perform the following action:

- The protocol translator down-line loads several initial fonts for the DECwindows login display.

  Once this is done, the terminal displays the DIGITAL logo and DECwindows login box.
- Log in to the system. This action causes more fonts to be loaded, then the remote session starts up.

Note: Due to heavy font usage, DECwindows applications may take longer than expected to start up when using XRemote. This additional time is only necessary at startup, to load the fonts. Once the application starts, performance and access times should be as expected.

Manual XRemote Session Startup

If you are not using a host computer that supports XDMCP or LAT, you must use manual session startup. Manual session startup involves several steps:

Step 1: Enabling XRemote manually on a terminal server port.

Step 2: Connecting to the host computer.

Step 3: Setting the location of the X display.

Step 4: Starting up client applications.

Step 5: Returning to the EXEC prompt.

Step 6: Enabling XRemote manually again on a terminal server port.

Enabling XRemote Manually

The EXEC command **xremote** prepares the protocol translator for your use of XRemote. The software replies with a message informing you of your X display location:

```
Xremote enabled; your display is darkstar:2018

Start your clients and type Xremote again
```

In the above message, the software informs you that your display is `darkstar:2018`. This information will be used in a following step.

Connecting to the Host Computer

Connect to the host computer using the **telnet**, **lat**, or **rlogin** commands, and log in normally.

Setting the Location of the X Display

Note: If you are using a version of Telnet (on the remote host) which supports the "X Display Location" option (RFC 1096), you may skip this step and proceed to the "Starting Up Client Applications" step.

Inform the host computer of the location of your X display, which was provided to you by the software when you entered the **xremote** command above.

For most versions of the UNIX operating system, the X display location is set by using the **setenv** command to set the environment variable DISPLAY. Refer to your UNIX systems' online, X(1) manual page for more information.

Example:

```
host_prompt% setenv DISPLAY darkstar:2018
```

On VAX/VMS, use the **SET DISPLAY** command. Refer to the *VMS DCL Dictionary* for more information.

Example:

```
$ SET DISPLAY/CREATE/NODE=DARKSTAR/SERVER=2018/TRANSPORT=[transport]
```

Note: You must install either the TCP/IP transport from DEC, or a third-party TCP/IP transport. Contact your VAX/VMS system administrator for the appropriate TCP/IP transport name.

Starting Up Client Applications

At this point, you must start your client applications for your host operating system.

The protocol translator accepts the X connection attempt from the client application, and places the client into a dormant state.

Returning to the EXEC Prompt

If it is possible to log off the host computer and keep your X clients running in the background, you can do so now. This conserves resources on the host and protocol translator that would otherwise be inaccessible until you exited from XRemote state.

If not, you may simply escape back to the protocol translator prompt using the escape character (Ctrl ^ X by default).

Re-enabling XRemote Manually

Enter the command **xremote** to access XRemote mode. If the X clients connected successfully, the session will be put into XRemote mode, and the clients will be allowed to complete their startup.

If no clients were found, you will see the following message:

```
No X clients waiting - check that your display is darkstar:2018
```

Check your hosts to determine if an error was made when starting the session. The most likely cause is an improperly specified display location. Another possible cause is the host computer does not recognize the name of the communications server through which you are attempting to make a connection.

Session Termination

In manual operation, the protocol translator and X terminal remains in XRemote mode until either all clients disconnect or a reset request is received from the X terminal.

If a session terminates during startup, it may be because you invoked transient X clients which set some parameters and disconnected (such as **xset** or **xmodmap**). There must always be one session connected or the connection will be reset.

X.25 Terminal Server Support

In addition to support of PAD-based X.25 connection service, the protocol translator also supports use of X.25 as a transport mechanism for IP-based traffic. This capability covers the Telnet **connect** and **rlogin** connection commands, but also extends to XRemote and TN3270. In addition, the protocol translator can route IP over X.25. LAT is not supported over X.25. Refer to the chapter "X.25 Configuration and Management" for complete details concerning configuration requirements for setting up X.25 encapsulation.

This capability is most useful on protocol translators with a serial network interface. From a user's point of view, connections are relatively transparent when making a remote connection over a PDN. In general, users simply make a connection request.

Example:

This example illustrates a connection attempt to an X.25 host named *far_host* using Telnet.

```
pt>telnet far_host
```

If properly configured, the protocol translator takes this connection request, encapsulates the IP-based frame in an X.25 packet and sends it over the X.25 PDN to a specified location (based on the X.121 address).

At the remote location, a Cisco router takes the request, strips encapsulation from the X.25 packet, and puts the IP-based frame on the appropriate network. A virtual terminal connection is then made between the originating node and the target host.

Note: As an alternative, use the protocol translator to support this requirement via X.25 PAD capabilities. Refer to the section "X.3 PAD Connections" in this chapter for more information.

Connection Management Commands

The following user-level commands and facilities apply to all transmission protocols supported by Cisco protocol translators:

- The connection escape sequence
- **resume** command
- **login** command
- **lock** command
- **where** command
- **name-connection** command
- **send** command
- **show sessions** command
- **show terminal** command
- **show users/systat** commands
- **exit**, **quit**, and **disconnect** commands

Escaping Out of a Connection

You can have several concurrent connections open and switch back and forth between them. The number of connections that can be opened is defined by the **session-limit** command, described in the chapter "System Configuration."

To switch between connections, first type the escape sequence, which by default is Ctrl ^ X (press the Ctrl, SHIFT and ^ key simultaneously, let go, then press the X key), to return to the system command prompt.

To make a new connection, use the procedures described in the preceding sections. To get back to an existing connection, use the **resume** command. Use the **where** command to list your open connections.

Resuming Previous Connections

The EXEC **resume** command returns to a previous connection. The general form is available with all connection protocols supported by the protocol translator. The general command syntax is as follows:

resume [*connection***]**

The optional argument *connection* is the connection name or number; the default is the most recent connection.

Note: See the sections "Switching Between Telnet Connections" and "The PAD Resume Command" for additional information about the **resume** command.

You can omit the command word **resume** and simply type the connection number to resume a connection. You can also return to the most recent session by just pressing the Return key.

Examples:

This command resumes connection 2.

```
PT>resume 2
```

You can omit the command name and simply type the connection number to resume that connection. This example resumes connection 3.

```
PT>3
```

To resume the most recent connection, simply press the Return key.

Changing a Login Name

Use the EXEC command **login** to log into a system with a specific user name. This command can be used to change your login name. You may choose to change your login name because an outgoing access list may expect a particular user name. (Refer to the sections about Cisco's TACACS and other user authentication facilities in the chapter "System Configuration." IP access lists are described in the chapter "TCP/IP Configuration and Management.")

login

The system returns prompts for user name and password. If both are entered correctly, your session becomes associated with the specified user name. If there is no match, your connection reverts to the user name with which the **login** command attempt was made, if applicable. If no login name and password were originally required, the connection reverts to a session that is not associated with any name.

Sample Session:

In this sample session, a user named *foouser* wants to change the login name being used to *sloan*. After entering the **login** command, the new name is entered along with an incorrect password. The system rejects the attempt to change the user name. Next, *foouser* attempts to change the login name to *klaus*. This time the correct password is entered and the user is now allowed access to the EXEC at the user-level under the user name of *klaus*.

```
PT>login

Username:sloan
Password:
% Access denied
```

```
Still logged in as "foouser"
PT>login

Username:klaus
Password:
PT>
```

Locking the Terminal

You can prevent access to your session while keeping your connection open by using the **lock** EXEC command. Enter this command at the EXEC prompt:

lock

This command locks your keyboard. You need to include the global configuration command **lockable** in your system configuration. This configuration command is described in the chapter "System Configuration."The **lock** EXEC command remains in effect until the **clear line** privileged EXEC command, described in the chapter "System Management," is executed.

When you lock a session, you are prompted for a password, which can be any arbitrary string. Enter the password you want to assign. The screen clears and displays the message "Locked." To regain access to your sessions, re-enter the password. The protocol translator honors session timeouts on a locked line.

Listing Open Connections

The EXEC command **where** displays information about all open LAT, Telnet, or rlogin connections associated with the current terminal line. Enter this command at the EXEC prompt:

where

Sample Session:

Following is a sample output from this command:

```
pt>where

Conn Host                 Address             Byte  Idle Conn Name
   1 MIS1               131.108.19.50          0     0   MIS1
*  2 OTTER              192.31.7.24            0     0   OTTER
```

The information displayed includes the host name, address, number of characters waiting to be sent to the terminal, idle time, and connection name. An asterisk (*) indicates the current connection.

Naming a Connection

The **name-connection** command assigns a logical name to a connection. The EXEC prompts for the connection number and name to assign when you enter this command. The **where** command displays a list of the assigned logical connection names.

Sample Session:

The following checks the connection number for the host *Eng1*, assigns the logical name *my host* to it, and then confirms the assignment.

```
pt>where
Conn Host     Address         Byte  Idle  Conn  Name
*  1 Eng1     192.31.6.22        0     0  Eng1
   2 Term2    192.33.6.21        0     0  Term2

pt>name-connection
Connection number:1
Enter logical name:  my host
Connection 1 to Eng1 will be named ”my host” [confirm]

pt>where
Conn Host     Address         Byte  Idle  Conn  Name
*  1 Eng1     192.31.6.22        0     0  my host
   2 Term2    192.33.6.21        0     0  Term2
```

Sending Messages to Other Terminals

To send messages to one or all terminal lines, use the privileged **send** EXEC command. Enter this command at the EXEC prompt:

send {*line-number* **|** ***}**

A common way of using this command is to inform users of an impending shutdown.To send a message to a particular line, use the argument *line-number* to specify the line. To send a message to all lines, use an asterisk (*). The system prompts for the message, which may be several text lines long. End the message by typing the Ctrl-Z key sequence. Type Ctrl-C to abort the command.

Sample Session:

This sample session illustrates how to send a message to all terminal users.

```
PT> send all
Enter message, end with CTRL/Z; abort with CTRL/C:
System shutdown in 10 minutes.
<Ctrl-Z>
Send message? [confirm] yes

***
***
*** Message from tty56, cube CE10, to all terminals:
***
System shutdown in 10 minutes.
```

Displaying Active Terminal Sessions

To display information about your active terminal sessions, enter the **show sessions** EXEC command at the system prompt:

show sessions

Sample Session:

The following illustrates how to obtain information about active sessions.

```
pt>show sessions

Conn Host                Address            Byte  Idle Conn Name
*  1 GUN                 131.222.3.11        0     0   GUN
   2 BIG                 131.222.3.14        0     5   BIG
```

The information displayed includes the host name, address, number of characters waiting to be sent to the terminal, idle time, and connection name. An asterisk (*) indicates your current terminal session.

Displaying Current Terminal Parameters

The EXEC command **show terminal** displays the configuration parameter settings for the current terminal line. Enter this command at the EXEC prompt:

show terminal

Sample Session:

Following is a sample display of the command's output:

```
pt>show terminal

Line 3, Location: "Library x1234", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600, no parity, 2 stopbits, 8 databits
The escape character is "^^", followed by "x"
The local hold character is disabled
No flowcontrol in effect.
Status: Ready, Active
Capabilities: none
Idle EXEC timeout is 10 minutes.
Idle session timeout is not set.
Modem answer timeout is 15 seconds
Dispatch timeout is not set.
Allowed transports are telnet rlogin.  Preferred is telnet
Disconnect character is not set
Activation character is ^M (13)
No output characters are padded
Characters causing immediate data dispatching:
   Char    ASCII
```

The display includes a comprehensive report on the terminal settings in effect, including the preferred transport protocol.

Displaying Line Information

The EXEC commands **show users** and **systat** display information about the active ports of the protocol translator. The commands are synonymous and have this syntax:

show users [all]
systat [all]

The information displayed includes the line number, connection name, idle time, and terminal location. The optional keyword **all** requests information for both active and inactive ports.

Sample Session:

Following is a sample display of the command's output:

```
pt>systat all

    Line    User      Host(s)             Idle   Location
   0 con 0                                       otter console
   1 tty 1                                       Bill  x1234
   2 tty 2            DAVE-SS2            1:32   Dave x1235
*  3 tty 3            STUFT               0      Denise x1236
   4 tty 4            incoming            11:17  remote.host.com
   5 tty 5                                       Sam- ROM emulator
   6 tty 6                                       CD4 ???
   7 tty 7                                       Teresa x1236
  10 tty 10           STUFT               3      Mark x1237
  11 tty 11            HEAT               1      Eng32-1
  12 tty 12           STFUT               17:32  CD8 ???
  13 tty 13                                      1525: Sandy x1238
  14 tty 14           MEDDLE               23    Marsha x1239
  15 tty 15                                      Randy x1240
  16 tty 16                                      Kevin  x2120
  17 tty 17                                      Robert x2141
  20 tty 20                                      Bill  x2142
  21 tty 21                                      Laser printer
  22 tty 22                                      ""
```

The information displayed includes the line number, connection name, idle time, and terminal location. The optional keyword **all** requests information about both active and inactive ports.

Exiting a Session

The **exit** or **quit** commands terminate the EXEC and close any active terminal sessions. The commands are synonymous; enter either command when you are finished with your session.

Disconnecting

The EXEC command **disconnect** closes a connection, and has this syntax:

disconnect [*connection***]**

The optional argument *connection* is a connection name or number; the default is the current connection. To list your open connection, use the **where** command.

Sample Session:

This sample illustrates how to disconnect a specific session.

```
pt>where

Conn Host     Address         Byte  Idle  Conn  Name
*  1 Eng1     192.31.6.22        0     0  my host
   2 Term2    192.33.6.21        0     0  Term2

pt>disconnect my host
Closing connection to Eng1 [confirm]
```

Note: Do not use the **disconnect** command to end a session. Instead, log off the host, thus allowing the host to initiate the disconnect then use **quit** or **exit** to close the session. If you cannot log off the host using **quit** or **exit**, then use the **disconnect** command.

Changing Terminal Parameters

This section describes how to locally change the terminal parameters using the EXEC **terminal** commands. The new settings temporarily override those made with the line sub-commands, remaining in effect only until the user exits the system.

Most **terminal** commands can be executed at the user-level prompt (>). To obtain information about the current terminal configuration parameter settings, enter the **show terminal** EXEC command. Settings can be changed or removed by using the keyword **no**; the command descriptions indicate the command syntax.

To display a list of supported **terminal** commands, enter the **terminal ?** EXEC command.

Some **terminal** commands use the decimal representation of an ASCII character as an argument. Refer to the appendix “ASCII Character Set” for an ASCII chart.

Recording the Terminal Type

To record the current terminal type, enter the **terminal terminal-type** EXEC command.

> **terminal terminal-type** *terminal-name*
> **terminal no terminal-type**

The argument *terminal-name* records the type of current terminal. The information can be passed as information to a remote host. For example, the *terminal-name* is used by TN3270 for display management, and by Telnet and rlogin to inform the remote host of the terminal type.

Use the **terminal no terminal-type** command to remove the terminal type.

Example:

This example sets the terminal type to VT-220.

```
PT>terminal terminal-type vt-220
```

Changing the Terminal Screen Length

To set the number of lines on the current terminal screen, enter the **terminal length** EXEC command.

> **terminal length** *screen-length*

The argument *screen-length* is the desired number of lines. The protocol translator uses this value to determine when to pause during multiple-screen output. The default length is 24 lines. A value of zero disables pausing between screens of output.

The screen length specified can be learned by remote hosts. For example, the rlogin protocol uses the argument *columns* to set up terminal parameters on a remote UNIX host.

Example:

This example disables pausing between screen output.

```
PT>terminal length 0
```

Changing the Terminal Screen Width

To set the number of columns on the terminal screen, enter the **terminal width** EXEC command.

> **terminal width** *columns*

The argument *columns* is the desired number of columns. The default is 80 columns.

The column width specified can be learned by remote hosts. For example, the rlogin protocol uses the argument *columns* to set up terminal parameters on a remote UNIX host.

Example:

This example sets the terminal character columns to 132.

```
PT>terminal width 132
```

Changing the Terminal Escape Character

To set the escape character for the current terminal line, enter the **terminal escape-character** EXEC command.

terminal escape-character *decimal-number*
terminal no escape-character

The argument *decimal-number* is the ASCII decimal representation of the desired escape character or a control sequence (Ctrl-P, for example). Typing the escape character followed by the X key returns you to the EXEC when you are connected to another computer. The default escape characters are Ctrl-^.

The **terminal no escape-character** command makes the Break key function as the escape sequence; Break need not be followed by the X key. However, you cannot set the console escape character to Break, because the operating system software interprets Break as an instruction to halt the system.

Refer to Table 3-12 for a list of escape character sequences supported by the protocol translator.

Example:

This example sets the escape character to Ctrl-P (ASCII decimal 17).

```
PT>terminal escape-character 17
```

Changing the Hold Character

To set, change, or remove the hold character, enter the **terminal hold-character** EXEC command.

terminal hold-character *decimal-number*
terminal no hold-character

The argument *decimal-number* is either the ASCII decimal representation of the desired hold character or else a control sequence (for example, Ctrl-C).

Typing the hold character temporarily halts the output at the terminal. To continue the output, type any other character. To send the hold character to the host, precede it with the escape character. By default, no local hold character is set.

The **terminal no hold-character** command clears the hold character.

Note: You cannot suspend output on the console terminal.

Example:

This example removes the previously set hold character.

```
PT>terminal no hold-character
```

Changing the Terminal Parity

To change the terminal parity, enter the **terminal parity** EXEC command.

terminal parity {none | even | odd | space | mark}

The **terminal parity** command defines the generation of the parity bit for the current terminal line. The default setting is **none**.

Changing the Terminal Baud Rate

To set the transmit and receive speeds of the current terminal line, enter the **terminal speed** EXEC command.

terminal speed *baud*

The argument *baud* can be: 300, 1200, 2400, 4800, 9600, 19200, or 38400. It can also be set to any user-defined speed in a range from 75 to 57600. The default speed is 9600 baud.

Note: The value specified in the **extra-baudrate** global configuration command replaces the value 38400 as a possible argument in the **terminal speed** EXEC command.

Example:

This example sets the baud rate to 19,200.

```
PT>terminal speed 19200
```

Changing the Data Bits

To set the number of data bits per character for the current terminal line, enter the **terminal databits** EXEC command.

terminal databits {5|6|7|8}

If parity is being specified, specify 7 data bits per character. If no parity generation is in effect, specify 8 data bits per character. The default is 8 data bits per character.

Changing the Stop Bits

To set the number of stop bits transmitted per byte by the current terminal line, enter the **terminal stopbits** EXEC command.

terminal stopbits {1|1.5|2}

The default is 2 stop bits.

Setting Terminal Flow Control

To set up the method of data flow control for the current terminal line, enter the **terminal flowcontrol** EXEC command.

terminal flowcontrol {none|software [in|out]|hardware}

The keyword **software** sets software flow control. An additional keyword specifies the direction: **in** causes the protocol translator to listen to flow control from the attached device, and **out** causes the protocol translator to send flow control information to the attached device. If you do not specify a direction, both directions are assumed.

Note: For software flow control, the default stop and start characters are Ctrl-S and Ctrl-Q (XOFF and XON). You can change them with the **terminal stop-character** and **terminal start-character** commands described later in this section.

The keyword **hardware** sets hardware flow control. For information about setting up the RS-232 line, see the hardware manual for your Cisco product.

By default, no flow control method is set for a line. This default is returned with the **none** keyword.

Example:

This example sets incoming software flow control.

```
PT>terminal flowcontrol software in
```

Changing the Start Character

To set or remove the character that signals the start of data transmission when software flow control is in effect, enter the **terminal start-character** EXEC command.

terminal start-character *decimal-number*
terminal no start-character

The argument *decimal-number* is the ASCII decimal representation of the desired start character. The default start character is Ctrl-Q (ASCII character 17).

Use the **terminal no start-character** command to remove the start character.

Example:

This example removes the start character.

```
PT>terminal no start-character
```

Changing the Stop Character

To change the character that signals the end of data transmission when software flow control is in effect, enter the **terminal stop-character** EXEC command.

terminal stop-character *decimal-number*
terminal no stop-character

The argument *decimal-number* is the ASCII decimal representation of the desired stop character. The default stop character is Ctrl-S (ASCII character 19).

Use the **terminal no stop-character** command to remove the stop character.

Example:

This example changes the stop character to Ctrl-O (ASCII decimal character 15).

```
PT>terminal stop-character 15
```

Changing the Character Padding

To set or remove character padding on the current terminal line, enter the **terminal padding** EXEC command.

terminal padding *decimal-number count*
terminal no padding *decimal-number*

The command sets padding for a specified output character. The argument *decimal-number* is the ASCII decimal representation of the character, and can be any of the 127 ASCII characters. The argument *count* is the number of NULL bytes sent after that character, up to 255 padding characters in length.

The **terminal no padding** command ends this padding after the character represented by *decimal-number.*

Example:

This example pads Ctrl-D (ASCII 4) with 164 NULL bytes.

```
PT>terminal padding 4 164
```

Changing the End-of-Line Characters

The **terminal telnet-transparent** command causes the current terminal line to send a Return (CR) as a CR followed by a NULL instead of a CR followed by a Line Feed (LF). This scheme permits interoperability with different interpretations of end-of-line handling in the Telnet protocol specification. To set or remove this scheme, enter one of these commands at the EXEC prompt:

terminal telnet-transparent
terminal no telnet-transparent

Setting the Packet Dispatch Character

To define or remove a character or string that causes a packet to be sent, enter the **terminal dispatch-character** EXEC command.

terminal dispatch-character *decimal-number1* **[***decimal-number2 … decimal-numberx***]**
terminal no dispatch-character

This command causes the protocol translator to attempt to buffer characters into larger-sized packets for transmission to the remote host. The protocol translator normally dispatches each character as it is typed.

The argument *decimal-number* is the ASCII decimal representation of the character or string; any number of characters can be entered.

Example:

This examples defines the string “Escape C I” as the dispatch character.

```
PT>terminal dispatch-character 27 67 73
```

Establishing Input Notification

To establish or remove input notification, enter the **terminal notify** EXEC command.

terminal notify
terminal no notify

When you have multiple concurrent connections, you may want to know when output is pending on a connection other than the current connection. For example, you may want to know when another connection receives mail or a message. The **terminal notify** command causes the protocol translator to notify you of pending output. The **terminal no notify** command ends such notifications.

Selecting File Download Mode

To temporarily set or remove the ability of a line to act as a transparent pipe for file transfers, enter the **terminal download** EXEC command.

terminal download
terminal no download

Use the **terminal download** command when running a program such as KERMIT, XMODEM, or CrossTalk that downloads a file across a line. This command sets up the terminal line as a transparent pipe that can be used to transmit data, and is equivalent to entering all the following commands:

```
terminal telnet-transparent
terminal no escape-character
terminal no hold-character
terminal no pad 0
  .
  .
  .
terminal no pad 128
terminal parity none
terminal databits 8
```

The **terminal no download** command restores the line’s original parameter settings.

Selecting the Preferred Terminal Transport Protocol

To select a preferred remote terminal protocol, enter the **terminal transport** EXEC command.

terminal transport preferred {telnet|pad|lat|rlogin|none}

This EXEC command sets the preferred protocol for the duration of the current session. For details on the parameters of this command, refer to “Defining the Transport Protocol for a Specific Line” in the chapter “System Configuration.”

Sample Session 1:

The following session makes a connection using Telnet, after rlogin is refused as a preferred protocol by the remote host *Eng2*:

```
PT>terminal transport preferred rlogin
PT>Eng2
Trying Eng2.Cisco.com (121.108.145.12)...
PT>Eng2
Trying Eng2.Cisco.com (121.108.145.12)...Open
login:
```

Sample Session 2:

The following example sets the preferred transport to **none** and then attempts to make a connection without specifying a protocol. A Telnet connection is made successfully once **telnet** is specified.

```
PT>terminal transport preferred none
PT>Eng2
% unknown command “Eng2”
PT>telnet Eng2
Trying Eng2.Cisco.com (121.108.145.12)...Open
login:
```

Displaying Debug Messages on the Console and Terminals

To set or remove the ability to copy **debug** command output and system error messages to the current terminal, enter the **terminal monitor** EXEC command.

terminal monitor
terminal no monitor

To use this command, you must first issue the **enable** command and enter the password to access the privileged command mode.

The DEC MOP Server

All Cisco internetworking products include a server that implements a subset of DEC’s Maintenance Operation Protocol (MOP) for Ethernet interfaces. The MOP server supports the request ID message, periodic system ID messages, and the remote console carrier functions.

The MOP server periodically multicasts a system ID message, which is used by DEC’s Ethernet configurator to determine what stations are present in an Ethernet network. The configurator is controlled by the Network Control Program (NCP) command **define module configurator**. For more information on this command, consult DECnet or VAX documentation.

The Cisco internetworking products use a MOP communication device code of 121. This code has been assigned to Cisco by DEC, although some versions of DECnet-VAX software may report the code numerically, rather than with a device name. The DEC Ethernet product also makes use of system ID messages received when building network maps.

The MOP server supports the DEC remote console function. Using this capacity, a system manager on a DECnet system can create a virtual terminal connection to a Cisco protocol translator. Only a single inbound connection per Ethernet interface is supported. The MOP protocol does not contain the necessary mechanisms for supporting more than one connection at a time.

MOP is *not* a routable protocol. To bridge the MOP console carrier and system ID functions, you must enable bridging for protocol type 6002. The periodic system ID messages are sent to the multicast address AB00.0002.0000.

The EXEC command **debug mop** reports interesting events occurring within the MOP server, including reception of request ID messages, transmission of system ID messages, and reservation and release of the remote console.

Enabling MOP for an Interface

To control whether MOP is enabled for an interface, use the **mop enabled** interface subcommand. The command syntax is:

mop enabled
no mop enabled

The default is enabled. Use the **no mop enabled** command to disable the MOP if you do not want to run it.

Controlling the MOP System ID Messages

To control whether MOP periodic system ID messages are sent out to an interface, use the **mop sysid** interface subcommand. The command syntax is:

mop sysid
no mop sysid

You can still run MOP without having the background system ID messages sent out. This lets you use the MOP remote console, but does not generate messages used by the configurator.

Use the **no mop sysid** command to disable MOP from sending the system ID messages.

Interface Subcommand Summary

This section provides a list of the interface subcommands described in this chapter, in alphabetic order.

[no] mop enabled

Controls whether MOP is enabled for an interface or disabled. The default is enabled. Use the **no mop enabled** to disable the MOP if you do not want to run it at all.

[no] mop sysid

Controls whether MOP periodic system ID messages are sent out to an interface. Use the **no mop sysid** to disable the MOP from sending the sysid messages.

User EXEC Commands Summary

Following is an alphabetically arranged summary of the protocol translator user EXEC commands.

[connect|telnet] *host* **[***port***]** **[**/*keyword***]**

Enter either **connect** or **telnet** with a host name to make a TCP Telnet connection, or just enter a host name or Internet address. The argument *host* is a host name or Internet address. The argument *port* is an optional TCP port number; default is Telnet server port 23. The optional /*keyword* is one of following:

- /**route** *path*—Specifies loose source routing; *path* is a list of host names or Internet addresses, ending with the destination.
- /**line**—Enables Telnet line (local edit) mode.
- /**debug**—Enables Telnet message mode.
- /**stream**—Enables raw TCP stream mode for connecting to nonTelnet ports.

disconnect

Use only when the **exit** or **quit** commands do not terminate a connection.

exit
quit

Terminates the EXEC and closes any active terminal session.

lat *name* **[node** *nodename* **| port** *portname* **| /debug]**

Connects to a learned service on a host running DEC LAT. Argument *name* specifies a LAT learned service. The keyword and argument **node** *nodename* specifies a connection to a LAT node. The keyword and argument **port** *portname* specifies a destination LAT port name. The keyword **/debug** enables LAT message mode.

lock

Locks access to the terminal by requesting a temporary password that must be re-entered to access the terminal.

login

Allows you to change your login name by prompting for a new user name and password.

name-connection

Displays prompts to rename a connection number. The name is displayed by the **where** command.

pad {*X.121-address* | *hostname***} [/cud** *text***] [/debug] [/reverse]**

Begins connection to an X.3 PAD host. The argument *X.121-address* is the X.121 address; use *hostname* to specify a defined X.25 host name. argument **/cud** *text* is a switch that includes *text* in the Call User Data field of the outgoing Call Request Packet. The keyword **/debug** is a switch that causes the informational level of logging messages to be printed whenever the remote host changes an X.3 parameter setting or sends any other X.29 control packet. The keyword **/reverse** is a switch that causes Reverse Charge calls to be accepted on a per-call, rather than a per-interface, basis.

resume [*connection***]**
<Return>

Resumes the previous connection unless the optional *connection* argument is entered to resume a specific connection number or name. Pressing the Return key also resumes the previous connection. Use the **where** command to display open connections.

resume [*connection***] [***/keyword***]**

Resumes previous Telnet connections.Argument *connection* is the connection name or number; the default is the most recent connection.

The argument */keyword* is one of the following:

- **/line**—Enables Telnet line mode.
- **/noline**—Disables Telnet line mode and enables character-at-a-time mode (default).
- **/debug**—Enables Telnet debugging mode.
- **/nodebug**—Disables Telnet debugging mode (default).
- **/stream**—Turns on stream processing, which enables a raw TCP stream with no Telnet control sequences.
- **/nostream**—Turns off stream processing, which enables the Telnet protocol (default).
- **/echo**—Enables local echoing of characters (default).
- **/noecho**—Disables local echoing of characters.

resume [*connection***] [/***keyword***] [/set** *parameter:value***]**

Resumes PAD connections and allows changing local X.3 parameters during a connection.The argument *connection* is the connection number. The argument */keyword* is one of:

- **/line**—Enables line-mode editing.
- **/noline**—Disables line mode and enables character-at-a-time mode, which is the default.
- **/debug**—Displays informational messages whenever the remote host changes an X.3 parameter or sends an X.29 control packet.
- **/nodebug**—Disables the informational display, which is the default.
- **/echo**—Enables local echo, which is the default.
- **/noecho**—Disables local echo.
- **/set**—Sets the X.3 PAD parameters defined by *parameter* and *value*, separated by a colon.

The X.3 parameters default to the following for outgoing connections: 2:1, 3:2, 4:1, 7:4, 16:127, 17:21, 18:19.

For incoming PAD connections, software sends an X.29 SET PARAMETER packet to set only the following parameters: 2:0, 4:1, 7:21, 15:0.

rlogin *host* **[debug]**

Connects to a UNIX host using the rlogin protocol.The argument *host* is a host name or Internet address. The keyword **debug** enables rlogin message mode.

send {*line-number* | *****}**

Sends message to other terminals. The optional argument *line-number* specifies a particular line. To send a message to all lines, use an asterisk (*). The system prompts for the message, which may be several text lines long. End the message by typing the Ctrl-Z key sequence. Type Ctrl-C to abort the command.

show sessions

Lists all open connections, including connection number, host address, and name. Same as entering the **where** command.

show terminal

Displays local terminal parameter settings.

{show users | systat}

Either command displays list of users currently logged in, along with host name, line number, and location.

tn3270 *hostname*

Begins a terminal emulation session with a remote IBM host using the TN3270 application. The argument *hostname* is the name of the IBM host.

where

Lists all open connections. The list includes connection number, host address, and name.

xremote lat *service*

Startup command for host supporting DECwindows login session. The argument *service* is the DEC LAT service name.

xremote xdm *hostname*

Startup command for host supporting XDMCP. The argument *hostname* is the name of host computer.

Terminal Parameter-Setting Commands Summary

Following is a summary, arranged alphabetically, of the terminal parameter setting commands that locally effect the terminal settings.

terminal databits {5|6|7|8}

Specifies data bits per character. Default is 8 bits per character.

terminal [no] dispatch-character *decimal-number1* **[***decimal-number2 … decimal-numberx***]**

Defines or removes a character or string that causes a packet to be sent. The argument *decimal-number* is the ASCII decimal representation of the character or string of characters.

terminal [no] download

Sets or removes the ability of a line to act as a transparent pipe that can transmit data when running a program such as KERMIT, XMODEM, or CrossTalk that downloads a file across a protocol translator line.

terminal [no] escape-character *decimal-number*

Sets the escape character for the current terminal line. The argument *decimal-number* is the ASCII decimal representation of the desired escape character or an escape sequence (Ctrl-P, for example). Default is Ctrl-X

terminal flowcontrol {none|software [in|out]|hardware}

Sets the method of data flow control for the current terminal line. The keyword **software** sets software flow control; keyword **in** causes the protocol translator to listen to flow control from the attached device, and keyword **out** causes the protocol translator to send flow control information to the attached device. If not set, both directions are assumed. The keyword **hardware** sets hardware flow control. The keyword **none** sets no flow control (default).

terminal [no] hold-character *decimal-number*

Sets, changes, or removes the hold character for the current terminal line. The argument *decimal-number* is either the ASCII decimal representation of the desired hold character or a control sequence (for example, Ctrl-C). By default, no local hold character is set.

terminal length *screen-length*

Sets the number of lines on the screen of the current terminal. The argument *screen-length* is the desired number of lines. The default length is 24 lines. A value of zero disables pausing between screens of output. The screen length specified can be learned by hosts.

terminal [no] monitor

Enables/disables logging of system debugging and event messages on the current terminal.

terminal [no] notify

Establishes/removes incoming message notification.

terminal [no] padding *decimal-number count*

Sets or removes the character padding on the current terminal line. The argument *decimal-number* is the ASCII decimal representation of the character. It can be any of the 127 ASCII characters, up to 255 padding characters in length. The argument *count* is the number of NULL bytes sent after that character.

terminal parity {none even|odd|space|mark}

Defines the generation of the parity bit for the current terminal line. The default setting is **none**.

terminal speed *baud*

Changes the transmit and receive speeds of the current terminal line. When used, argument *baud* is typically specified as 300, 1200, 2400, 4800, 9600, 19200, or 38400, but can be set to any user-defined speed in a range from 75 to 57600. The default speed is 9600 baud.

terminal [no] start-character *decimal-number*

Sets or removes the character that signals the start of data transmission when software flow control is in effect. The argument *decimal-number* is the ASCII decimal representation of the desired start character. The default start character is Ctrl-Q (ASCII character 17).

terminal stopbits {1|1.5|2}

Sets the number of stop bits transmitted per byte by the current terminal line. The default is 2 stop bits.

terminal [no] stop-character *decimal-number*

Sets or removes the character that signals the end of data transmission when software flow control is in effect. The argument *decimal-number* is the ASCII decimal representation of the desired stop character. The default stop character is Ctrl-S (ASCII character 19).

terminal [no] telnet-transparent

Sets or removes ability of the terminal line to send a Return (CR) as a CR followed by a NULL instead of a CR followed by a Line Feed (LF). This scheme permits interoperability with different interpretations of end-of-line handling in the Telnet protocol specification.

terminal [no] terminal-type *terminal-name*

Records, removes, or changes the current terminal type. The type in the argument *terminal-name* is passed to applications that set terminal types.

terminal transport [telnet|lat|pad|rlogin|none]

Sets the preferred protocol for the duration of the current session. Default is LAT.

terminal width *columns*

Sets or changes the number of columns on the current terminal screen. The default is 80 columns.