Terminal Services Configuration Guide, Cisco IOS XE Release 16.x

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

Terminal Services Configuration Guide, Cisco IOS XE Release 16.x
© 2019 Cisco Systems, Inc. All rights reserved.

CONTENTS

Terminal Services Overview

This chapter provides an overview of Cisco IOS terminal services and includes the following main sections:

Cisco IOS Network Access Devices

Network devices that support access services enable single users to access network resources from remote sites. Remote users include corporate telecommuters, mobile users, and individuals in remote offices who access the central site. Access services connect remote users over serial lines to modems, networks, terminals, printers, workstations, and other network resources on LANs and WANs. In contrast, routers that do not support access services connect LANs or WANs.

Note Access services are supported on the Cisco 2500, Cisco 2600, and Cisco 3600 series routers. See the *Cisco Products Quick Reference Guide*, available at Cisco.com, for more information about Cisco devices for terminal and modem access services.

Figure 1 illustrates the following access services available in the Cisco IOS software:

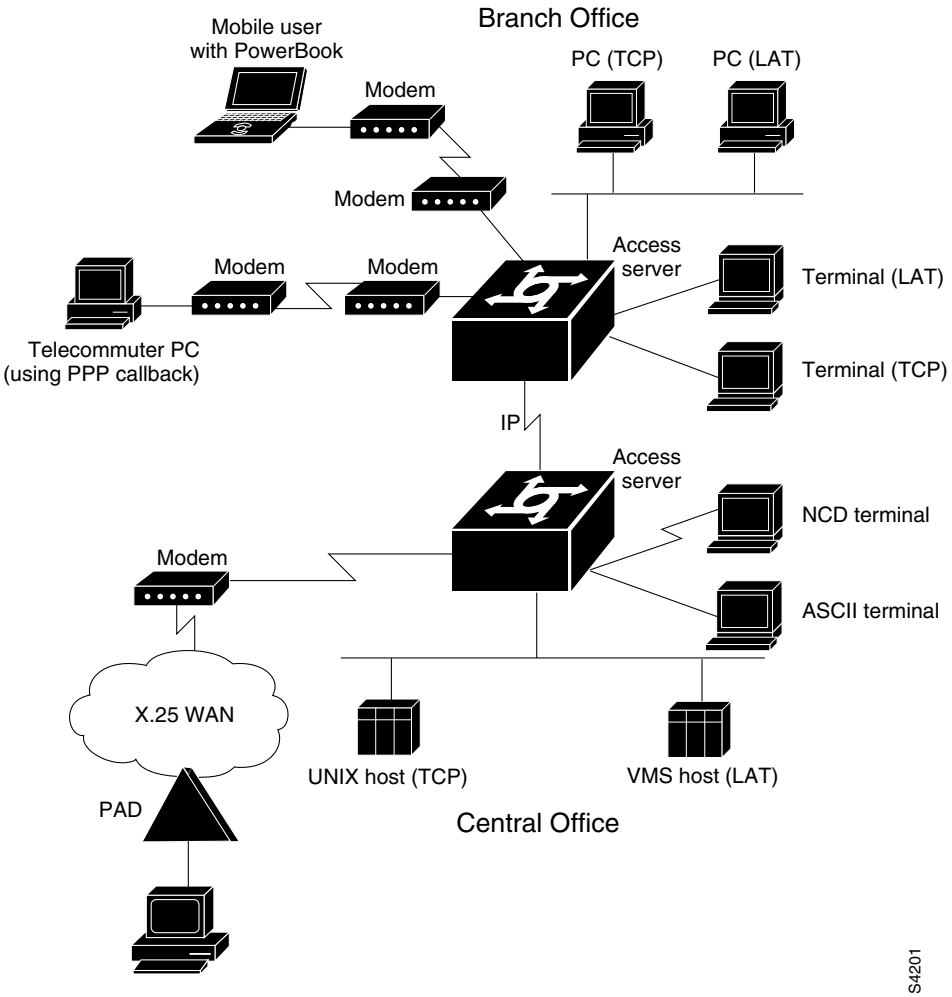
- Terminal services are shown between the terminals and hosts running the same protocol (LAT to LAT or TCP to TCP).
- Protocol translation is supported between the terminals and hosts running unlike protocols (such as LAT to TCP or TCP to LAT).

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

Note Protocol translation on Cisco4000 Series ISRs is supported only between TCP to X25 and X25 to TCP. Other protocols such as PPP, LAT, SLIP are not supported.

Asynchronous IP routing is shown by the PC running Serial Line Internet Protocol (SLIP) or Point-to-Point Protocol (PPP), and between the two access servers. Asynchronous routing configuration is described in the *Cisco IOS Terminal Services Configuration Guide*, Release 12.2.

Figure 1 Access Service Functions



Line Characteristics and Modems

The Cisco IOS software permits you to connect to asynchronous serial devices such as terminals and modems and to configure custom device operation. You can configure a single physical or virtual line or a range of lines. For example, you can configure one line for a laser printer and then configure a set of lines to switch incoming modem connections to the next available line. You also can customize your configurations. For example, you can define line-specific transport protocols, control character, and packet transmissions, set line speed, flow control, and establish time limits for user access.

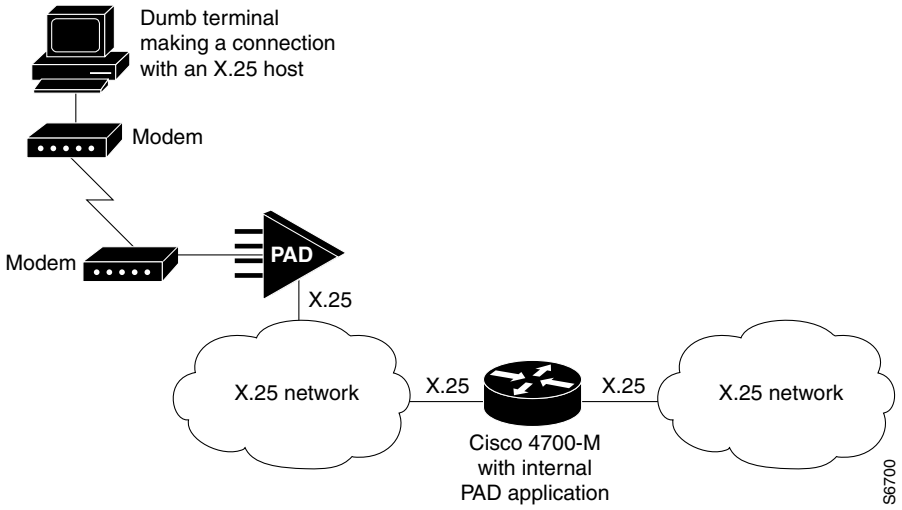
The chapters in this publication describe how to configure the lines for a specific device application. See the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" in this publication, and the chapters "Interfaces, Controllers, and Lines Used for Dial Access Overview" and "Preparing Modem and Asynchronous Interfaces" in the *Cisco IOS Dial Technologies Configuration Guide* for additional information about configuring Cisco asynchronous serial interfaces.

Asynchronous Character Stream Calls

Asynchronous character stream calls enter the router or access server through virtual terminal (vty) lines and virtual asynchronous interfaces (vty-async). These virtual lines and interfaces terminate incoming character streams that have no physical connection to the access server or router (such as a physical serial interface). For example, if you begin a PPP session over an asynchronous character stream, a vty-async interface is created to support the call. The following types of calls are terminated on a virtual asynchronous interface: Telnet, local-area transport (LAT), V.120, TN3270, and Link Access Procedure, Balanced-terminal adapter (LAPB-TA) and packet assembler/disassembler (PAD) calls.

Figure 2 shows a dumb terminal using a modem and packet assembler/disassembler (PAD) to place a call in to an X.25 switched network. The Cisco 4700-M router is configured to support vty lines and vty-async interfaces.

Figure 2 Standard X.25 Dial-Up Connection



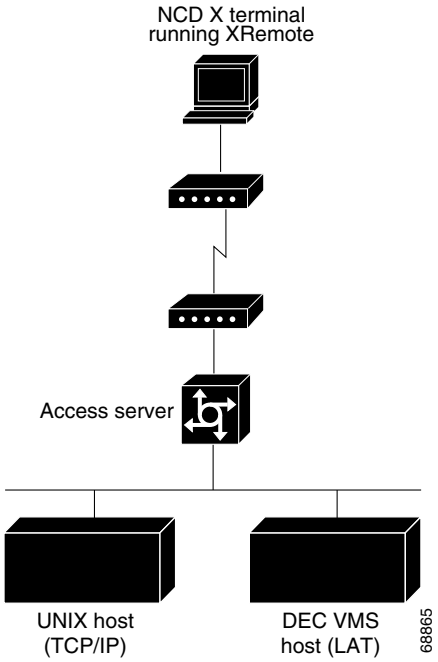
Remote Node Services

Remote node services permit remote users to connect devices over a telephone network using the following protocols:

- XRemote, the Network Control Device, Inc. (NCD) X Window Systems terminal protocol, which is described in the section "Configuring XRemote" in the "Configuring Dial-In Terminal Services" chapter in this publication.

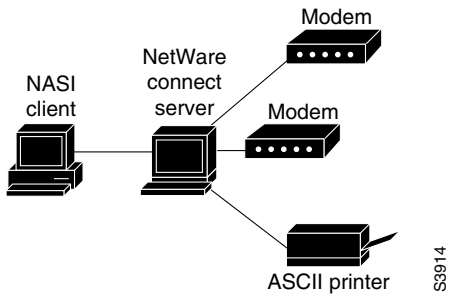
Remote users with X terminals, such as NCD terminals, use the XRemote protocol over asynchronous lines. The router provides network functionality to remote X terminals. Figure 3 illustrates an XRemote connection.

Figure 3 XRemote Connection



- NetWare Access Server Interface (NASI) server, which is described in the chapter "Configuring Support for NASI Clients to Access Network Resources" in this publication. Configuring a NASI server enables NASI clients to connect to asynchronous resources attached to a router. NASI clients are connected to the Ethernet interface 0 on the router. When the user on the NASI client uses the Windows or DOS application to connect to the router, a list of available terminal and virtual terminal lines appears. The user selects the desired outgoing terminal and virtual terminal port. (See Figure 4.)

Figure 4 NASI Setup in a NetWare Environment



Terminal Services

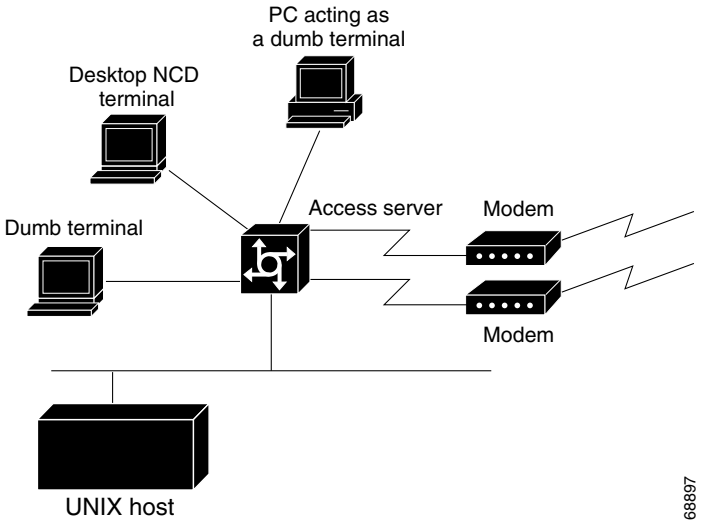
Terminal services permit asynchronous devices to be connected to a LAN or WAN through network and terminal-emulation software including Telnet, rlogin, NASI, the Digital local-area transport (LAT) protocol, and IBM TN3270. (See Figure 5.)

Access services permit terminals to connect with remote hosts using virtual terminal protocols including Telnet, NASI, LAT, TN3270, rlogin, and X.25 packet assembler/disassembler (PAD). You can use a router that supports access services to function as a terminal server to provide terminal access to devices on the network.

A host can also connect directly to an access server. In IBM environments, TN3270 allows a standard ASCII terminal to emulate a 3278 terminal and access an IBM host across an IP network.

In Digital environments, LAT support provides a terminal with connections to VMS hosts. X.25 PAD allows terminals to connect directly to an X.25 host over an X.25 network through the router. X.25 PAD eliminates the need for a separate PAD device. This connection requires use of one of the synchronous serial interfaces on the router supporting access services.

Figure 5 Terminal-to-Host Connectivity



Protocol Translation

Protocol translation services are essentially an extension of terminal services. A user running a TCP/IP-based application can connect to a host running a different virtual terminal protocol, such as the Digital LAT protocol. The Cisco IOS software converts one virtual terminal protocol into another protocol. Protocol translation enables users to make connections to X.25 machines using X.25 PAD.

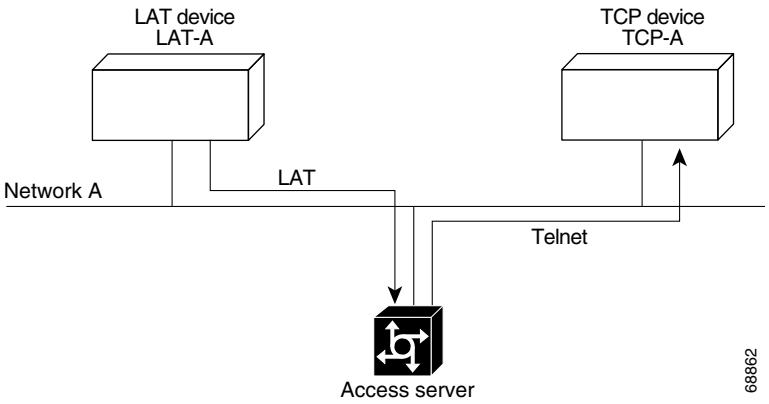
Note In Cisco4000 Series ISRs, translations is supported only between X25 and TCP, The other protocols that are described in this section is only for informational purpose.

Routers translate virtual terminal protocols to allow communication between devices running different protocols. Protocol translation supports Telnet (TCP), LAT, and X.25. One-step protocol translation software performs bidirectional translation between any of the following protocols:

- X.25 and TCP
- X.25 and LAT
- LAT and TCP

Figure 6 illustrates LAT-to-TCP protocol translation.

Figure 6 LAT-to-TCP Protocol Translation



Connecting to IBM hosts from LAT, Telnet, rlogin, and X.25 PAD environments requires a two-step translation process. In other words, users must first establish a connection with the router, then use the TN3270 facility to make a connection to the IBM host.

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

Configuring Terminal Operating Characteristics for Dial-In Sessions

This chapter describes how to set operating characteristics for remote terminal service connections. It includes the following main sections:

- Terminal Operating Characteristics Overview
- Selecting a Preferred Connection Protocol
- Configuring Communication Parameters for Terminal Ports

For a complete description of the terminal characteristic commands in this chapter, refer to the *Cisco IOS Terminal Services Command Reference*, Release 12.2. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the "Using Cisco IOS Software" chapter.

Terminal Operating Characteristics Overview

In line configuration mode, you can set terminal operating characteristics that will be in operation for that line until the next time you change the line parameters. Alternatively, you can change the line setting locally (temporarily) with **terminal** EXEC commands. Both tasks are described in this chapter.

Selecting a Preferred Connection Protocol

Your first task is to select a preferred connection protocol, then configure the appropriate communication parameters. The preferred transport type is your preferred connection protocol. To configure the router to support specific protocols, perform the tasks described in the following sections:

- Specifying the Transport Protocol
- Specifying a Local Transport Protocol

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

Specifying the Transport Protocol

Use the **transport preferred** command to specify which transport protocol is used on connections. Use the **transport input** and **transport output** commands to explicitly specify the protocols allowed on individual lines for both incoming and outgoing connections.

Note Cisco routers do not accept incoming network connections to asynchronous ports (TTY lines) by default. You must specify an incoming transport protocol before the line will accept incoming connections. For example, if you are using your router as a terminal server to make console-port connections to routers or other devices, you will not be able to use Telnet to connect to these devices. You will receive the message "Connection Refused."

For routers that support the Digital local-area transport (LAT) protocol, the default protocol for outgoing connections is LAT. For those that do not support LAT, the default protocol for outgoing connections is Telnet. For incoming connections, all the supported network protocols are accepted.

To specify transport protocols, use one or more of the following commands in line configuration mode:

Command	Purpose
`Router(config-line)# transport input {lat | mop | nasi | none | pad | rlogin | ssh | telnet | v120}`	Defines which protocols can be used to connect to a specific line.
`Router(config-line)# transport output {lat | mop | nasi | none | pad | rlogin | telnet | v120}`	Determines the protocols that can be used for outgoing connections from a line.
`Router(config-line)# transport preferred {lat | mop | nasi | pad | rlogin | telnet | v120}`	Specifies the protocol for the router to use if the user did not specify a protocol.
`Router(config-line)# transport preferred none`	Prevents errant connection attempts.

The IOS software accepts a host name entry at the EXEC system prompt as a Telnet command. If you enter the host name incorrectly, the router interprets the entry as an incorrect Telnet command and provides an error message indicating that the host does not exist. The **transport preferred none** command disables this option so that if you enter a command incorrectly at the EXEC prompt, the software does not attempt to make a Telnet connection to a host that it cannot find.

The **transport preferred** command setting specifies a search order when attempting to resolve names that might be valid for multiple protocols. If the address or service does not match the preferred protocol, all other valid output protocols are searched to find a valid match.

Specifying a Local Transport Protocol

You can configure the Cisco IOS software to save local parameters between sessions. These local parameters are set with **terminal** EXEC commands.

To specify the preferred protocol to use for the current session when a command does not specify one, use the following command in EXEC mode:

Command	Purpose
`Router> terminal transport preferred {lat | mop | nasi | none | pad | rlogin | telnet | v120}`	Specifies the protocol for the Cisco IOS software to use for the current session if the user did not specify a protocol.

The preferred transport type is your preferred connection protocol. This setting specifies a protocol search order that the Cisco IOS software uses when it attempts to resolve a device name that you enter, but you do not specify a connection protocol. For example, if you want to connect to a TCP/IP host named host1 and want to use Telnet, you enter the **telnet host1** command. However, if your preferred connection protocol is set to Telnet, you could enter only the **host1** argument and be connected to the device. A host name might be valid for multiple protocols. If the address or service does not match the preferred protocol, all other valid connection protocols are searched to find a valid match for the name.

For router software images that support LAT, the default protocol for outgoing connections is LAT. For router software images that do not support LAT, the default protocol for outgoing connections is Telnet. For incoming connections, all the supported network protocols are accepted.

The Cisco IOS software accepts a host name entry at the EXEC prompt as a Telnet command. If you enter the host name incorrectly, the Cisco IOS software interprets the entry as an incorrect Telnet command and provides an error message indicating that the host does not exist. The **transport preferred none** command disables this option so that if you enter a command incorrectly at the EXEC prompt, the Cisco IOS software does not attempt to make a Telnet connection.

Configuring Communication Parameters for Terminal Ports

To configure communication parameters, perform the tasks described in the following sections:

- Configuring Sessions on a Line (Required)
- Configuring Local Session Parameters (As Required)
- Changing the Default Privilege Level for Lines (As Required)
- Enabling Password Checking at Login (As Required)
- Establishing Terminal Session Limits (As Required)
- Displaying Line Connection Information After the Login Prompt (As Required)

Configuring Sessions on a Line

The Cisco IOS software supplies the following default serial communication parameters for terminal and other serial device operation:

- 9600 bits per second (bps) line speed
- 8 data bits
- 2 stop bits
- No parity bit

To change the default parameters as necessary to meet the requirements of the terminal or host to which you are connected, use any of the following commands in line configuration mode:

Command	Purpose

`Router(config-line)# speed bps`
or
`Router(config-line)# txspeed bps`
or
`Router(config-line)# rxspeed bps`
	Sets the line speed. Choose from line speed, transmit speed, or receive speed.

`Router(config-line)# databits {5 | 6 | 7 | 8}`	Sets the data bits.
`Router(config-line)# stopbits {1 | 1.5 | 2}`	Sets the stop bits.
`Router(config-line)# parity {none | even | odd | space | mark}`	Sets the parity bit.

Configuring Local Session Parameters

To change these parameters as necessary to meet the requirements of the terminal or host to which you are attached, use the following commands in EXEC mode, as needed:

Command	Purpose

`Router> terminal speed bps`
or
`Router> terminal txspeed bps`
or
`Router> terminal rxspeed bps`
	Sets the line speed for the current session. Choose from line speed, transmit speed, or receive speed.

`Router> terminal databits {5 | 6 | 7 | 8}`	Sets the data bits for the current session.
`Router> terminal stopbits {1 | 1.5 | 2}`	Sets the stop bits for the current session.
`Router> terminal parity {none | even | odd | space | mark}`	Sets the parity bit for the current session.

Changing the Default Privilege Level for Lines

To change the default privilege level for a given line or a group of lines, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# privilege level level`	Specifies a default privilege level for a line.

Enabling Password Checking at Login

You can enable password checking on a particular line so that the user is prompted to enter a password at the system login screen. You must then also specify a password. To do so, use the following commands in line configuration mode:

	Command	Purpose
Step 1	`Router(config-line)# login`	Enables password checking on a per-line basis using the password specified with the **password** command.
Step 2	`Router(config-line)# password password`	Assigns a password to a particular line.

You can enable password checking on a per-user basis, in which case authentication is based on the username specified with the **username** global configuration command. To enable password checking on a per-user basis, use the following commands in line configuration mode:

	Command	Purpose
Step 1	`Router(config-line)# login local`	Enables password checking on a per-user basis using the username and password specified with the **username** global configuration command.

Step 2	`Router(config-line)# login tacacs`
	or
	`Router(config-line)# login authentication {default | list-name}`
		Selects the TACACS style user ID and password-checking mechanism.

Use the **login tacacs** command with TACACS and extended TACACS. Use the **login authentication** command with AAA/TACACS+.

By default, virtual terminals require passwords. If you do not set a password for a virtual terminal, the router displays an error message and closes the attempted connection. Use the **no login** command to disable this function and allow connections without a password.

For other access control tasks and password restrictions, including the **enable password** global configuration command that restricts access to privileged mode, see the *Cisco IOS Security Configuration Guide*, Release 12.2.

Establishing Terminal Session Limits

You might need to control terminal sessions in high-traffic areas to provide resources for all users. You can define the following limitations for terminal sessions:

- The maximum number of sessions
- The session timeout interval

To establish terminal session limits, use the following commands in line configuration mode:

	Command	Purpose
Step 1	`Router(config-line)# session-limit session-number`	Sets the maximum number of simultaneous sessions.[1]

	Command	Purpose

Step 2	`Router(config-line)# session-timeout minutes [output]`
	or
	`Router(config-line)# absolute-timeout minutes`
		Sets an idle timeout interval on a console or terminal (tty) line.
		Sets a timeout interval on a virtual terminal (vty) line.

Step 3	`Router(config-line)# logout-warning [seconds]`	Warns users of impending timeouts set with the **absolute-timeout** command.

1. There is no inherent upper limit to the number of sessions you can create.

The **session-timeout** command behaves slightly differently on virtual (vty) terminals than on physical console, auxiliary (aux), and terminal (tty) lines. When a timeout occurs on a vty, the user session returns to the EXEC prompt. When a timeout occurs on physical lines, the user session is logged out and the line returned to the idle state.

The **absolute-timeout** command terminates the connection after the specified time period has elapsed, regardless of whether the connection is being used at the time of termination. You can specify an absolute-timeout value for each port. The user is given 20 seconds notice before the session is terminated. You can use this command along with the **logout-warning** command, which notifies the user of an impending logout.

You can use a combination of the **exec-timeout** line configuration command, which sets the interval that the EXEC command interpreter waits until user input is detected, and the **session-timeout** line configuration command, both set to approximately the same values, to get the same behavior from virtual lines that the **session-timeout** command causes on physical lines.

The **absolute-timeout** command overrides any timeouts set through the AppleTalk Remote Access (ARA) protocol.

Displaying Line Connection Information After the Login Prompt

You can display the host name, line number, and location of the host each time an EXEC session is started or an incoming connection is made. The line number banner appears immediately after the EXEC banner or incoming banner. This feature is useful for tracking problems with modems because it lists the host and line for the modem connection. Modem type information is also included if applicable.

To provide line information after the login prompt, use the following command in global configuration mode:

Command	Purpose
`Router(config)# service linenumber`	Provides service line number information after the EXEC banner or incoming banner.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

Configuring Dial-In Terminal Services

This chapter describes how to configure support for asynchronous character stream calls running Telnet, rlogin, local-area transport (LAT), XRemote, or TN3270. It includes the following main sections:

For a complete description of the dial-in terminal services commands in this chapter, refer to the *Cisco IOS Terminal Services Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

Dial-In Terminal Service Overview

Inbound asynchronous character stream calls are routed to virtual terminal lines and virtual asynchronous interfaces, which are used to terminate incoming character steams that do not share a physical connection with the access server or router (such as a physical interface). A virtual asynchronous interface is the place where inbound Telnet, LAT, V.120, TN3270, and packet assembler/disassembler (PAD) calls or sessions terminate on the router. Virtual terminal lines are used for attaching to the router in a nonphysical way.

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

Configuring support for terminal service connections means enabling network devices running the same protocol to connect across a LAN or WAN through network and terminal-emulation software.

The following sections describe how to configure these supported dial-in terminal services:

- Configuring Telnet and rlogin—Of all protocol suites, TCP/IP is the most widely implemented on networks of all media types. TCP/IP is the current standard for internetworking and is supported by most computer vendors, including all UNIX-based workstation manufacturers. TCP/IP includes Telnet and rlogin.
- Connecting a VMS Host Using LAT—The proprietary LAT terminal connection protocol from Digital Equipment Corporation used with Digital minicomputers.
- Configuring TN3270—IBM 3278 terminal emulation provides TN3270-based connectivity to IBM hosts over serial lines.
- Configuring XRemote—The X Window Systems terminal protocol from Network Control Devices, Inc., provides network functionality to remote X terminals.

Each section provides examples of how to configure and connect to a terminal service.

Configuring Telnet and rlogin

Telnet and rlogin are protocols that enable TCP/IP connections to a host. Telnet, a virtual terminal protocol that is part of the TCP/IP protocol suite, is the more widely used protocol. The rlogin protocol is a remote login service developed for the Berkeley Software Distribution (BSD) UNIX system. It provides better control and output suppression than Telnet, but can only be used when the host (typically, a UNIX system) supports rlogin. The Cisco IOS implementation of rlogin does not subscribe to the rlogin “trusted host” model. That is, a user cannot automatically log in to a UNIX system from the router, but must provide a user ID and a password for each connection.

Telnet allows a user at one site to establish a TCP connection to a login server at another site, then passes the keystrokes from one system to the other. Telnet can accept either an IP address or a domain name as the remote system address. In short, Telnet offers three main services:

- Network virtual terminal connection
- Option negotiation
- Symmetric connection

The Cisco implementation of Telnet supports the following Telnet options:

- Remote echo
- Binary transmission
- Suppress go ahead
- Timing mark
- Terminal type
- Send location
- Terminal speed
- Remote flow control
- X display location

Telnet and rlogin Configuration Task List

To configure Telnet and rlogin, perform the tasks in the following sections:

- Configuring Telnet and UNIX rlogin (Required for Service)
- Making Telnet and UNIX rlogin Connections (Required for Making Connections)
- Using UNIX Style Syntax for rlogin Connections (Optional)

The section "Monitoring TCP/IP Connections" later in this chapter provides tasks for maintaining TCP/IP connections.

Configuring Telnet and UNIX rlogin

To configure support for Telnet or rlogin calls, use the following commands beginning in line configuration mode.

Command	Purpose
`Router(config-line)# telnet speed default-speed maximum-speed`	Negotiates speeds on reverse Telnet lines.
`Router(config-line)# telnet refuse-negotiations`	Causes Telnet to refuse to negotiate full-duplex, remote echo requests on incoming connections.
`Router(config-line)# telnet transparent`	Sets line to send a RETURN (CR) as a CR followed by a NULL instead of a CR followed by a LINE FEED (LF).
`Router(config-line)# telnet sync-on-break`	Sets the line to send a Telnet SYNCHRONIZE signal when it receives a Telnet BREAK signal.
`Router(config-line)# telnet break-on-ip`	Sets the line to cause the system to generate a hardware BREAK signal on the EIA/TIA-232 line that is associated with a reverse Telnet connection when a Telnet Interrupt-Process command is received on that connection.
`Router(config)# ip tcp chunk-size number`	In global configuration mode, optimizes the line by setting the number of characters output before the interrupt executes.
`Router(config-if)# ip alias ip-address tcp-port`	In interface configuration mode, assigns an IP address to the service provided on a TCP port.
`Router(config)# busy-message hostname d message d`	In global configuration mode, defines a message that the router displays whenever a Telnet or rlogin connection to the specified host fails.
`Router(config)# login-string hostname d message [%secp] [%secw] [%b] d [%m] d`	In global configuration mode, defines a message that the router displays whenever a Telnet or rlogin connection to the specified host succeeds.
`Router(config-line)# notify`	Sets up a line to notify a user that has multiple, concurrent Telnet connections when output is pending on a connection other than the current one.
`Router(config-line)# refuse-message d message d`	Defines a "line-in-use" message to indicate that the line is currently busy.

The **telnet speed** command sets the line speed to match line speeds on remote systems in reverse Telnet, on host machines hooked up to an access server or router to access the network, or on a group of console lines hooked up to the access server or router when disparate line speeds are in use at the local and remote ends of the connection. Line speed negotiation adheres to the Remote Flow Control option, defined in RFC 1080.

The **telnet refuse-negotiations** command suppresses negotiation of the Telnet Remote Echo and Suppress Go Ahead options.

The **telnet transparent** command is useful for coping with different interpretations of end-of-line handling in the Telnet protocol specification.

The **telnet sync-on-break** command sets the line to cause a reverse Telnet line to send a Telnet SYNCHRONIZE signal when it receives a Telnet BREAK signal. The Telnet SYNCHRONIZE signal clears the data path, but the line still interprets incoming commands.

Enter the **telnet break-on-ip** command to control the translation of Telnet Interrupt-Process commands into X.25 BREAK indications, and to work around the following situations:

- Several user Telnet programs send a Telnet Interrupt-Process command, but cannot send a Telnet BREAK signal.
- Some Telnet programs implement a BREAK signal that sends a Telnet Interrupt-Process command.
- Some EIA/TIA-232 hardware devices use a hardware BREAK signal for various purposes.

When the **telnet break-on-ip** command is used with a correctly operating host, Cisco IOS software implements the Telnet SYNCHRONIZE and ABORT OUTPUT signals, which can stop output within one packet worth of data from the time the user types the interrupt character. Enter the **ip tcp chunk-size** command to configure a faster response to user interrupt characters. Changing the number of characters output, or chunk size, affects neither the size of the packet used nor the TCP window size, either of which would cause serious efficiency problems for the remote host and for the access server or router. Instead, the system software checks the Telnet status after the number of characters specified, causing only a relatively minor performance loss.

Use the **ip alias** command to configure connections to an IP address to act identically to connections made to the primary IP address of the server on the TCP port. A user trying to connect is connected to the first free line in a rotary group using the Telnet protocol.

With the **login-string** command options, you can set a pause, prevent a user from issuing commands during a pause, send a BREAK character, and use a percent sign (%) in the login string. The **busy-message** command and **login-string** command are only useful with two-step protocol translation sessions. For more information about protocol translation, see the chapter “Configuring Protocol Translation and Virtual Asynchronous Devices” in this publication.

For actual sample configurations on how to configure Telnet and rlogin, see the section “Telnet and rlogin Examples” later in this chapter.

Making Telnet and UNIX rlogin Connections

To provide Telnet and rlogin connection capabilities, use the following commands in EXEC mode:

	Command	Purpose

Step 1	`Router> connect host [port] [keyword]`
	or
	`Router> telnet host [port] [keyword]`
		Logs in to a host that supports Telnet. Refer to the descriptions for the **connect** and **telnet** commands in the *Cisco IOS Terminal Services Command Reference*, for a list of supported keywords.[1]

Step 2	`Router> show hosts`	Displays a list of available hosts.
Step 3	`Router> show tcp`	Displays the status of all TCP connections.
Step 4	**Ctrl^**	Logs out of the host by entering the default escape sequence.[2]
Step 5	Choose from the following list of escape sequences, according to your task: Press **Ctrl^ b** if your task is to break. Press **Ctrl^ c** if your task is to interrupt a process (IP). Press **Ctrl^ h** if your task is to erase a character (EC). Press **Ctrl^ o** if your task is to abort an output display (AO). Press **Ctrl^ t** if your task is to confirm you are at the host. Press **Ctrl^ u** if your task is to erase a line (EL).	Logs out of the host by entering a special escape sequence.[2] These special Telnet sequences map generic terminal control functions to operating system-specific functions.
Step 6	**Ctrl^ ?**	Lists the available Telnet commands at any time during the active Telnet session.[2]
Step 7	**exit** or **logout**	Exits a Telnet or rlogin session.

1. Cisco IOS software provides a robust collection of connection options. The options allow for enhanced sessions allowing, for example, encrypted sessions, Kerberos login, and File Transfer Protocol and World Wide Web connections. Additionally, it is possible to suppress system messages, including IP addresses and server names, displayed during session connection and disconnection. This function allows transparent TCP connections and can be useful when an asynchronous tunnel connection is being made.
2. Press and hold the **Ctrl** and **Shift** keys while pressing the **6** key. You can enter the command character as you hold down the **Ctrl** key or with **Ctrl** released; you can enter the command characters as either uppercase or lowercase letters.

With the Cisco IOS implementation of TCP/IP, you are not required to enter the **connect** or **telnet** commands to establish a Telnet connection. You can just enter the learned host name as long as the host name is different from a command word for the router. Telnet must be the default (you can make it the default with the **transport preferred** command). Use the **show hosts** EXEC command to display a list of the available hosts. Use the **show tcp** EXEC command to display the status of all TCP connections. The Cisco IOS software assigns a logical name to each connection, and 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 **name-connection** EXEC command. If the name is already in use, the Cisco IOS software assigns a null name to the connection. For an example of making a Telnet connection, see the section "Telnet and rlogin Examples" later in this chapter.

After you enter the **rlogin** command, you can have several concurrent rlogin connections open and switch between them. To open a new connection, exit the current connection by entering the escape sequence (**Ctrl-Shift-6** then **x** [**Ctrl^x**] by default) to return to the system command prompt, then open a new connection. For an example of making an rlogin connection or switching between connections, see the sections "rlogin Connection Example" or "Switch Between Telnet and rlogin Sessions Example" later in this chapter.

Note We recommend that you use Encrypted Kerberized Telnet whenever you establish a Telnet session to a router or access server, which protects the integrity of the device. For information about Encrypted Kerberized Telnet, refer to *Cisco IOS Security Configuration Guide*.

Using UNIX Style Syntax for rlogin Connections

The **rlogin** command supports the standard BSD UNIX **-l** option. Before this addition was introduced, the **rlogin** command allowed remote users to log in using the **/user** *username* option, which was not compatible with the standard UNIX **rlogin -l** *username* option.

This feature is supported on all of Cisco TCP/IP-enabled routers and access servers.

To set up this UNIX feature, use one of the following the following commands in EXEC mode:

Command	Purpose
`Router# rlogin hostname`	Enters the name of the host to which you are connecting.
`Router# rlogin hostname [-l hostname] [/user hostname]`	Enters the user name.
`Router# rlogin hostname [-l hostname] [/user hostname] debug`	(Optional) Enters the **debug** mode to troubleshoot the connection from the remote site to the host.
`Router# rlogin hostname [-l hostname] [/user hostname] /quiet`	(Optional) Enters the **/quiet** keyword to make a transparent connection from the remote site to the host.

When you are done with the UNIX session, use the **exit** command to end it.

Monitoring TCP/IP Connections

To display the status of a TCP connection or view a summary of the TCP connection endpoints in the system, use the following commands in user EXEC mode:

Command	Purpose
`Router> show tcp [line-number]`	Displays the status of a TCP connection.
`Router> show tcp brief [all]`	Displays a summary of the TCP connection endpoints in the system.

Telnet and rlogin Examples

This section provides the following examples:

- Telnet Connection Example
- Telnet Connection Without and With Messages Suppressed Example
- rlogin Connection Example
- rlogin UNIX-Style Syntax Example

- Switch Between Telnet and rlogin Sessions Example
- List Supported Telnet Commands Example

Telnet Connection Example

The following example establishes a telnet connection to a host named server1 and specifies vt100 as the terminal type for the session:

Router> **telnet server1 /terminal-type vt100**

The following example connects to a host with logical name host1:

```
Router> host1
```

Telnet Connection Without and With Messages Suppressed Example

The following examples show how to suppress the onscreen messages displayed during login and logout of a Telnet session.

The following example shows the messages displayed when a connection is made *without* using the optional **/quiet** keyword with the **telnet** EXEC command to suppress messages from the operating system:

```
Router# telnet Server3

Translating "Server3"...domain server (172.18.89.42) [OK]
Trying Server3--Server3.cisco.com (172.18.89.42)... Open
Kerberos:       No default realm defined for Kerberos!


login: User2
Password:
        Welcome to OpenVMS VAX version V6.1 on node CRAW
    Last interactive login on Tuesday, 15-DEC-1998 11:01
    Last non-interactive login on Sunday,  3-JAN-1999 22:32

Server3) logout
   User2        logged out at  16-FEB-2000 09:38:27.85
[Connection to Server3 closed by foreign host]
Router#
```

The following example shows the limited messages displayed when connection is made using the optional **/quiet** keyword:

```
Router# telnet Server3 /quiet

login: User2
Password:
         Welcome to OpenVMS VAX version V6.1 on node CRAW
     Last interactive login on Tuesday, 15-DEC-1998 11:01
     Last non-interactive login on Sunday,  3-JAN-1999 22:32

Server3) logout
    User2        logged out at  16-FEB-2000 09:38:27.85
Router#
```

The **/quiet** keyword is useful for making transparent connections during asynchronous tunnel connections. The keyword can be used with any of the EXEC connection commands—**connect**, **telnet**, and **rlogin**.

Note The Cisco IOS software offers the **ip telnet quiet** global configuration command, which also suppresses onscreen messages during Telnet connections. The **ip telnet quiet** command is set globally, and is useful to Internet service providers that want to permanently suppress onscreen system connection messages that often include information such as server names and IP addresses. Refer to the *Cisco IOS Dial Technologies Command Reference*, for more information about the **ip telnet quiet** command.

rlogin Connection Example

The following example makes an rlogin connection to a host at address 172.31.21.2 and enables the message mode for debugging:

```
Router> rlogin 172.31.21.2 debug
```

rlogin UNIX-Style Syntax Example

The following example illustrates how a user named jsmith can use the **rlogin ?** help command and the debug mode to establish and troubleshoot a remote connection to the host named Alviso:

```
Router> rlogin ?
  WORD IP address or hostname of a remote system
Router> rlogin Alviso ?
  -l     Specify remote username
  /user  Specify remote username
  debug  Enable rlogin debugging output
  <cr>
Router> rlogin Alviso -l ?
  WORD  Remote user name
Router> rlogin Alviso -l jsmith ?
  debug  Enable rlogin debugging output
  <cr>
Router> rlogin Alviso -l jsmith debug
```

Switch Between Telnet and rlogin Sessions Example

You can switch between sessions by escaping one session and resuming a previously opened session. The following example shows how to escape out of a connection to the host named host1 and to resume connection 2. You escape out of the current session and return to the EXEC prompt by entering the command sequence **Ctrl-Shift-6** then **x**. Resume the connection with the **resume** command.

```
host1% ^^X
Router> resume 2
```

You can omit the command name and simply enter the connection number to resume that connection. The following example illustrates how to resume connection 3:

```
Router> 3
```

To list all the open sessions associated with the current terminal line, use the **where** command.

List Supported Telnet Commands Example

At any time during an active Telnet session, you can list the Telnet commands by pressing the escape sequence keys (by default Ctrl-Shift-6) followed by a question mark at the system prompt:

```
Ctrl-^ ?
```

A sample of this list follows:

```
Router> ^^?

[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
```

Note In screen output examples that show two caret (^^) symbols together, the first caret represents the Ctrl key and the second caret represents the keystroke sequence Shift-6. The double caret combination (^^) means hold down the Ctrl key while you press the Shift and the 6 keys.

Using Cisco DialOut for Telnet Connections

The Cisco DialOut feature enables users on a workstation operating Windows to send faxes or connect to service provider services outside the LAN by using modems attached or internal to a network access server. The Cisco DialOut feature extends the functionality of Telnet by enabling users to control the activity of these modems from their desktop computers using standard communications software.

The Cisco DialOut feature has two components:

- Telnet Extensions for Dialout—Network access server component
- The DialOut Utility—Client/desktop component

Both components are required and neither can function as a stand-alone feature.

The Telnet Extensions for Dialout component uses reverse Telnet to access modems attached to the network access server. This component enables the network access server to interface with the client/desktop component of the Cisco DialOut feature and to return CARRIER DETECT signals to the communications software so that the software can determine when to start dialing a particular number.

Telnet extensions allow the communications software running on the desktop computer of the client to control modem settings such as baud rate, parity, bit size, and stop bits.

To enable this feature, you only need to configure the access server or router for reverse Telnet and configure the appropriate lines to send and receive calls.

The client/desktop component of Cisco DialOut feature must be installed on the client workstation before this feature can be used. For information about installing and using the client/desktop component of the Cisco Dial-Out feature, and configuring the access server, see the *DialOut Utility User Guide* Cisco publication at Cisco.com.

Configuring Stream TCP

Stream TCP connections, or raw TCP or TCP-Clear connections as they are sometimes called, are used to transport a stream of 8-bit characters as-is over an IP network, between a TCP client and TCP server system. This method is used to transport legacy asynchronous application data through an IP network, for example, with a Point-of-Sale (PoS) terminal connecting to an application server.

To establish a Stream TCP connection from an EXEC session, use the **/stream** keyword with the **telnet** command. You will also generally want to configure the line to provide for data transparency. See the following procedure for the steps to do this.

Stream TCP Autocommand Procedure

In the following procedure, a line is configured so that any connection into it is automatically connected using Stream TCP to the application server at the specified IP address and TCP port (IP address 10.1.2.3 and TCP port 4321 in the examples).

Step 1 Configure the line for data transparency using the following configuration as an example:

```
Router# configure terminal

Router(config)# line 33
Router(config-line)# no motd-banner
Router(config-line)# no exec-banner
Router(config-line)# no vacant-message
Router(config-line)# escape-character NONE
Router(config-line)# no hold-character
```

Step 2 Configure the autocommand:

```
Router(config-line)# autocommand telnet 10.1.2.3 4321 /quiet /stream
```

Step 3 Configure the **telnet-faststream** option (this is an optional step). On platforms that support this feature such as the Cisco AS5800 access servers, you may want to configure the **telnet-faststream autocommand** option to provide for Stream TCP performance enhancements. An example of how this option can be entered follows:

```
Router(config-line)# autocommand-options telnet-faststream
```

Connecting a VMS Host Using LAT

Connection to a VMS host is slightly different if you are connecting to a VMS host running VMS Version 5.4 or earlier than when connecting to a VMS host running VMS Version 5.5 or later software.

VMS Version 5.4 or Earlier System

If a host-initiated connection is received that specifies a destination port number that corresponds to a virtual port on the router, a virtual EXEC process will be created to allow the user to log in. This process can be used, in conjunction with the Digital **set host/dte** command on VMS, to connect to a router named router1 from a VMS host node, as shown in the following example:

```
$lcp :==$latcp
$lcp create port lta300:
$lcp set port lta300:/service=able /node=router1
$set host/dte lta300:
```

VMS Version 5.5 or Later System

To connect to a VMS host running VMS Version 5.5 or later software, you must turn on the outgoing connections of the VMS LAT hosts and use the Digital **set host/lat** command, as shown in the following example:

```
$lcp :== $latcp
$lcp set node/connection =outgoing
$set host/lat able
```

Port Names When Configuring a LAT Printer

When you configure a LAT printer, the LAT port name is the line number without a "TTY" designation on the **show lines** command output. For example, if you configure terminal line 10 (named ABLE) to be a LAT printer port, you must use the OpenVMS command to associate an arbitrary LAT device to the LAT port name, as follows:

```
$lcp :== $lcp
$lcp create port lta300:
$lcp set port/node=ABLE/port=10 lta300:
```

The LAT port name is the line number without the "TTY," regardless of whether the format of the TTY line number is decimal or octal.

Additional LAT Capability

The Cisco IOS software fully supports the LAT protocol suite, and provides the following features:

- High-speed buffering—Handles a full screen of data (2000 characters) at full speed without requiring additional flow control.
- Protocol transparency—Handles connections transparently. The user needs no protocol information to establish a connection.
- Simplified configuration management—Uses logical names for LAT group codes to simplify the network structure.
- Maintenance Operation Protocol (MOP)—Supports the Digital protocol to support the request ID message, periodic system ID messages, and the remote console carrier functions for Ethernet interfaces.

LAT Configuration Task List

The Cisco IOS software LAT protocol is supplied with a default configuration and does not require additional configuration for you to use it.

To enable LAT and customize LAT for your particular network environment, perform the tasks described in the following sections:

- Configuring Basic LAT Services (Required for Service)
- Enabling Inbound Services (As Required)
- Controlling Service Announcements and Service Solicitation (As Required)
- Configuring Traffic Timers (As Required)

- Optimizing Performance (As Required)
- Defining LAT Access Lists (As Required)
- Enabling Remote LAT Modification (As Required)
- Making LAT Connections (Required for Making Connections)

The section "Monitoring and Maintaining LAT Connections" later in this chapter provides tips for maintaining LAT connections. The section "LAT Configuration and Connection Examples" later in this chapter provides LAT configuration examples.

Configuring Basic LAT Services

To enable basic LAT services, use the following commands beginning in interface configuration mode:

	Command	Purpose
Step 1	`Router(config-if)# lat enabled`	Enables the LAT protocol. LAT is disabled by default.
Step 2	`Router(config-if)# lat node node-name`	Gives the router a LAT node name that is different than the host name.
Step 3	`Router(config-line)# lat out-group {groupname number | range | all}`	(Optional) Defines the group list for an outgoing connection on a specified line.
Step 4	`Router(config)# lat group-list groupname {number | range | all} [enabled | disabled]`	(Optional) Specifies logical names for group lists.
Step 5	`Router(config)# lat service-group {groupname | number | range | all} [enabled | disabled}`	(Optional) Specifies groups to be advertised.
Step 6	`Router(config-line)# lat remote-modification`	(Optional) Enables remote LAT modification of line characteristics.

Use the **lat out-group** command to define the list of services to which a user can connect. You create this list by defining the group code lists used for connections from specific lines. You can limit the connection choices for an individual line by defining the group code lists for an outgoing connection. When a user initiates a connection with a LAT host, the line of the user must share a common group number with the remote LAT host before a connection can be made.

Use the **lat group-list** command to specify a name for group lists to simplify the task of entering individual group codes. A name makes it easier to refer to a long list of group code numbers. To display the defined groups, use the **show lat groups** command.

Use the **lat service-group** command to specify a group code mask to use when advertising all services for a node. You can enter more than one group code by listing the numbers. You can also enter both a group code name and group codes.

Use the **lat remote-modification** line configuration command to configure a LAT line so that a remote LAT node can change the operating characteristics of the line.

Enabling Inbound Services

Just as LAT services are offered by host computers, they also can be offered by access servers and routers, because they implement both the host and server portions of the LAT protocol. This capability allows connections from either hosts or local access servers or routers. A host connected to a local device is called a *host-initiated connection*.

The tasks described in this section define support for host-initiated connections. This support includes refining the list of services that the router will support. An incoming session can be to either a port or a service. The port name is the terminal line number, as reported by the **show users all** EXEC command.

To enable inbound services, use the following commands in global configuration mode as needed:

Command	Purpose
`Router(config)# lat service service-name password password`	Sets the LAT password for a service.
`Router(config)# lat service service-name ident identification`	Sets the LAT service ID for a specific service.
`Router(config)# lat service service-name rating static-rating`	Specifies a static service rating for a specific service.
`Router(config)# lat service service-name rotary group`	Configures a LAT rotary group.
`Router(config)# lat service service-name autocommand command`	Associates a command with a specific service for auto-execution.
`Router(config)# lat service service-name enabled`	Enables inbound connections to a specific service.

Use the **show lat advertised** EXEC command to display LAT services offered to other systems on the network.

A service must be specifically enabled, but not all of the attributes in the previous task table are necessary in a particular environment.

Controlling Service Announcements and Service Solicitation

You can configure the Cisco IOS software to support the service responder feature that is part of the LAT Version 5.2 specification.

Specifically, the DECserver90L+, which has less memory than other Digital servers, does not maintain a cache of learned services. Instead, the DECserver90L+ solicits information about services as they are needed.

LAT Version 5.2 nodes can respond for themselves, but LAT Version 5.1 nodes, for example, VMS Version 5.4 or earlier nodes, cannot. Instead, a LAT Version 5.2 node configured as a service responder can respond in proxy for those LAT Version 5.1 nodes.

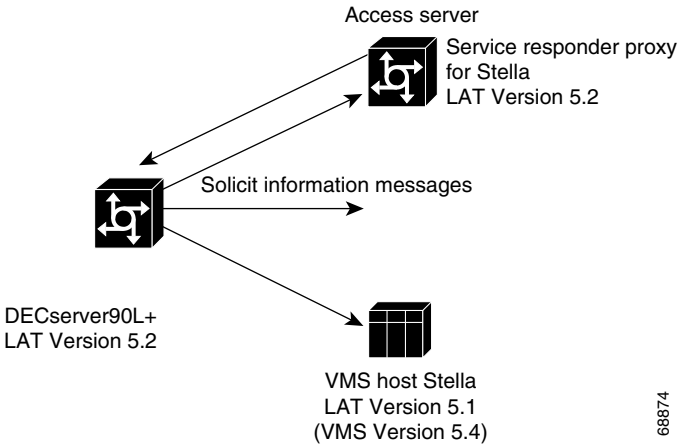
The Cisco IOS software can be configured as a LAT service responder. Of course, if all your nodes are LAT Version 5.2 nodes, you need not enable the service responder features.

To control service announcements and service solicitations, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# lat service-responder`	Enables a proxy node to respond to solicit-information multicast messages.
Step 2	`Router(config)# no lat service-announcements`	Disables periodic broadcasts of service advertisements.
Step 3	`Router(config)# lat service-timer interval`	Adjusts the time between service announcements.

Use the **lat service-responder** command to configure the Cisco IOS software to respond to solicit information requests addressed to LAT Version 5.1 nodes. This function allows nodes that do not cache service advertisements to interoperate with nodes that do not respond to solicit requests. Figure 1 shows how a router can act as a proxy for LAT servers.

Figure 1 Router as Proxy for LAT Server



The DECserver90L+ broadcasts a solicit information request in search of service for address Stella. The VMS host, Stella, is unable to respond to the request because it is running LAT Version 5.1. The access server is running LAT Version 5.2 with service responder enabled and informs the DECserver90L+ of the address for Stella.

Use the **no lat service-announcements** command to disable periodic broadcasts of service announcements. If service announcements are enabled, the LAT node will periodically broadcast service advertisements. If service announcements are disabled, the LAT node will not send service announcements, so a remote node requiring connection to the local node must use solicit-information messages to look up node information. Disable service announcements only if all of the nodes on the LAN support the service responder feature.

Use the **lat service-timer** command to adjust the time between LAT service advertisements for services offered. This command is useful in large networks with many LAT services and limited bandwidth.

Configuring Traffic Timers

You can customize the environment for sending LAT messages. The Cisco IOS implementation of LAT allows you to set the following features:

- The number of retransmissions before declaring a system unreachable
- The interval of time LAT waits before sending a keepalive message on an idle connection
- The interval of time LAT waits between transmission of messages

These features affect all LAT connection types.

To enable these features, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# lat retransmit-limit number`	Sets the message retransmit limit.
Step 2	`Router(config)# lat ka-timer seconds`	Sets the keepalive timer.
Step 3	`Router(config)# lat vc-timer milliseconds`	Sets the virtual circuit timer.

Optimizing Performance

To optimize performance for your LAT environment, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# lat vc-sessions number`	Sets the maximum number of sessions on a LAT virtual circuit. The maximum (and default) number of sessions is 255.
Step 2	`Router(config)# lat host-buffers receive-buffers`	Allows a LAT host node to receive more than one message at a time.
Step 3	`Router(config)# lat server-buffers receive-buffers`	Allows a LAT server node to receive more than one message at a time.
Step 4	`Router(config)# lat host-delay number`	Specifies the delay acknowledgment for incoming LAT slave connections, where *number* is milliseconds.

Use the **lat host-buffers** command to set the number of messages received by a host at one time. Increasing this number can enhance performance. Before LAT Version 5.2, LAT allowed only one outstanding message at one time on a virtual circuit. This restriction could limit the performance of the Cisco IOS software when it processed a large number of messages because only one Ethernet packet of data could be in transit at a time. During virtual circuit startup, each side communicates to the other how many outstanding messages it is willing to accept.

Use the **lat server-buffers** command to set the number of messages received by a server at one time. Increasing this number can enhance performance. Before LAT Version 5.2, LAT allowed only one outstanding message at one time on a virtual circuit. This restriction could limit the performance of Cisco IOS software when it processed a large number of messages because only one Ethernet packet of data could be in transit at a time. With LAT Version 5.2, nodes can indicate that they are willing to receive more than one message at a time. During virtual circuit startup, each side communicates to the other how many outstanding messages it is willing to accept.

Use the **lat host-delay** command to set a user-defined delay for the acknowledgment for incoming LAT slave connections. This command is useful in situations where you need to control the delay. For example, if data is being transferred between a Digital server (using LAT) and a UNIX host (using Telnet) via a protocol translator, the protocol translator imposes the LAT delay on the Telnet and the LAT service, where Telnet may time out due to the LAT restriction.

Defining LAT Access Lists

Because LAT groups were not intended to implement security or access control, the Cisco IOS software supports *access lists* to provide these functions. An access list is a sequential collection of permit and deny conditions that serve to restrict access to or from LAT nodes on a specific terminal line. Each access list statement defines a permit or deny condition and a matching criterion for the node name.

When a LAT connection is attempted (either incoming or outgoing), the node name of the destination service (*not* the service name) is compared against the regular expression. If they match, the connection is permitted or denied as specified.

To define access lists and conditions, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router# configure terminal`	Enters global configuration mode.
Step 2	`Router(config)# lat access-list` *number* {**permit** | **deny**} *node-name*	Specifies an access condition.
Step 3	`Router(config)# line` *line-number*	Enters line configuration mode.
Step 4	`Router(config-line)# access-class` *access-list-number* {**in** | **out**}	Restricts incoming and outgoing connections between a particular terminal line or group of lines and the node names in an access list.

Enabling Remote LAT Modification

You can configure a LAT line so that a remote LAT node can change the operating characteristics of the line. To enable remote LAT modification, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# lat remote-modification`	Enables remote LAT modification of line characteristics.

Making LAT Connections

The LAT protocol is most often used to connect routers to Digital hosts. LAT is a Digital-proprietary protocol, and the Cisco IOS software uses LAT technology licensed from Digital to allow the following LAT services:

- Make a LAT connection
- Define a group code list for outgoing LAT connections
- Switch between LAT sessions
- Use Digital commands on the server
- Exit a LAT session

For actual LAT connection examples, see the section "LAT Configuration and Connection Examples" later in this chapter.

To enable specific LAT connections or services, use the following commands in EXEC mode:

	Command	Purpose
Step 1	Router> **lat** *name* [**node** *node-name* | **port** *portname* | **/debug**]	Connects to a LAT host.[1]
Step 2	Router> **terminal lat out-group** {*groupname* | *number* | *range*}	(Optional) Defines a temporary list of services to which you or another user can connect by defining the group code lists used for connections from specific lines.
Step 3	Router> **show lat services** [*service-name*]	(Optional) Lists available LAT services.
Step 4	Router> **help**	(Optional) Lists the subset of Digital commands that the Cisco IOS software supports.

1. You can quit the connection by pressing **Ctrl-C** or complete the connection by entering the password for a given service.

You can also set your preferred connection protocol to any available connection protocol supported in the Cisco IOS software. Your preferred connection protocol is also referred to in the Cisco IOS software as a "preferred transport type." If your preferred connection protocol is set to **lat**, you can use the **connect** command in place of the **lat** command. To configure a preferred connection protocol, use the **transport preferred** command. When your preferred connection protocol is set to **none** or to another protocol, you must use the **lat** command to connect to a LAT host.

To specify a temporary list of services to which you or another user can connect, you must define the group code lists used for connections from specific lines. You limit the connection choices for an individual line by defining the group code lists for an outgoing connection. To define a group code list, use the **terminal lat out-group** command. When a user initiates a connection with a LAT host, the line of the user must share a common group number with the remote LAT host before a connection can be made. The group code range *must be* a subset of the configured group code range of the line.

You can have several concurrent LAT sessions open and switch between them. To open a subsequent session, first enter the escape sequence (Ctrl-Shift-6 then x [Ctrl^x] by default) to suspend the current session. Then open a new session. To list the available LAT services, enter the **show lat services** EXEC command.

When you are done with the LAT session, use the **exit** command to end it, then terminate the active LAT session by entering the Ctrl-C key sequence.

Monitoring and Maintaining LAT Connections

To monitor and maintain LAT connections, use the following commands in EXEC mode as needed:

Command	Purpose
Router> **clear entry** *number*	Deletes an entry from the queue.
Router> **show entry**	Displays queued host-initiated connections.
Router> **show lat advertised**	Displays LAT services offered to other LAT systems.
Router> **show lat groups**	Displays defined LAT groups.
Router> **show lat nodes**	Displays information about LAT nodes.

Command	Purpose
Router> **show lat services** [*service-name*]	Displays information about LAT learned services.
Router> **show lat sessions** [*line-number*]	Displays active LAT sessions.
Router> **show lat traffic**	Displays traffic and resource utilization statistics.
Router> **show node** [**all** | *node-name*] [**counters** | **status** | **summary**]	Displays information about LAT nodes. Information is displayed in the same way as in the Digital interface.
Router> **show service** [*service-name*]	Displays LAT learned services.

LAT Configuration and Connection Examples

This section provides the following LAT examples:

- Basic LAT Service Example
- LAT Service with Selected Group Codes Example
- Displaying LAT Services on the Same LAN Example
- Establishing an Outbound LAT Session Example
- Logically Partitioning LAT Services by Terminal Line Example
- LAT Rotary Groups Example
- Associating a Rotary Group with a Service Example
- LAT Access List Example
- LAT Connection Examples

Basic LAT Service Example

The following example establishes the LAT service named ABLE for your router. Subsequently, your router advertises ABLE (with default group code 0) on the LAN. Other LAT nodes can connect to you using LAT service ABLE, provided the group codes on the LAT nodes and the group codes for ABLE intersect. By default, most LAT nodes, such as OpenVMS Version 5.5 hosts, have user group code set to 0, so you have default access to ABLE.

```
! Create LAT service with password protection and
! identification string using the following global configuration commands.
 lat service ABLE password secret
 lat service ABLE ident Welcome to my machine
```

LAT Service with Selected Group Codes Example

The following example establishes the LAT service named ABLE from your router with selected group codes 1, 4 through 7, and 167. This configuration limits inbound access to those LAT nodes that have group codes that intersect with those for LAT service ABLE.

```
! Establish a LAT group list.
 lat group-list HUBS 1 4-7 167
```

```
!
! Enable LAT group list for the service-group.
 lat service-group HUBS enabled
!
! Create LAT service with password protection and
! identification string.
 lat service ABLE password secret
 lat service ABLE ident Welcome to my machine
```

Displaying LAT Services on the Same LAN Example

The following example demonstrates how you can check which LAT services are on the same LAN as your router. Note that the LAT service named ABLE is also listed, with the "Interface" column listing the interface as "Local."

```
Router> show lat services

Service Name      Rating    Interface     Node (Address)
CAD                16       Ethernet0     WANDER
ABLE               16       Local
CERTIFY            33       Ethernet0     STELLA
```

Establishing an Outbound LAT Session Example

The following example establishes a LAT session to remote LAT service HELLO using an interactive session:

```
Router> lat HELLO
```

Logically Partitioning LAT Services by Terminal Line Example

The following example illustrates how LAT services are logically partitioned by terminal line. At the example site, lines 1 through 7 go to the shop floor, lines 8 through 11 go to the Quality Assurance department, and lines 12 through 16 go to a common area.

```
! Define LAT groupnames.
 lat group-list DEFAULT 0
 lat group-list FLOOR 3
 lat group-list QA 4

line 1 7
 lat out-group FLOOR enabled
 lat out-group DEFAULT disabled
line 8 11
 lat out-group QA enabled
 lat out-group DEFAULT disabled
 line 12 16
lat out-group DEFAULT QA FLOOR enabled
```

LAT Rotary Groups Example

The following example illustrates how to configure a range of lines for rotary connections and then establishes the LAT service named Modems for rotary connection:

```
! Establish rotary groups.
```

```
line 3 7
 rotary 1
!
! Establish modem rotary service.
!
 lat service Modems rotary 1
 lat service Modems enabled
```

Associating a Rotary Group with a Service Example

The following example defines a service that communicates with a specific line and defines a rotary with only that line specified. You can establish rotary groups using line configuration commands and the **rotary** line configuration command.

```
hostname ciscots
! Service name for the access server as a whole.
lat service ciscopt enable
! Set up some lines with unique service names.
line 1
 rotary 1
 lat service ciscopt1 rotary 1
 lat service ciscopt1 enable
!
line 2
 rotary 2
 lat service ciscopt2 rotary 2
 lat service ciscopt2 enable
```

LAT Access List Example

The following example illustrates incoming permit conditions for all IP hosts and LAT nodes with specific characters in their names and a deny condition for X.25 connections to a printer. Outgoing connections, however, are less restricted.

```
! Permit all IP hosts, LAT nodes beginning with "VMS" and no X.25
! connections to the printer on line 5.
!
access-list 1 permit 0.0.0.0 255.255.255.255
lat access-list 1 permit ^VMS.*
x29 access-list 1 deny .*
!
line 5
 access-class 1 in
!
! Meanwhile, permit outgoing connections to various places on all the
! other lines.
!
! Permit IP access within cisco.
access-list 2 permit 172.30.0.0 0.0.255.255
!
! Permit LAT access to the Stella/blue complexes.
lat access-list 2 permit ^STELLA$
lat access-list 2 permit ^BLUE$
!
! Permit X25 connections to infonet hosts only.
x29 access-list 2 permit ^31370
!
line 0 99
 access-class 2 out
```

The following example illustrates how to define access lists that permit all connections, thereby conforming to software behavior prior to Cisco IOS Release 9.0. Remember that the value supplied for the *list* argument in both variations of the **access-class** commands is used for *all* protocols supported by the Cisco IOS software. If you are already using an IP access list, it will be necessary to define LAT (and possibly X.25) access lists permitting connections to all devices, to emulate the behavior of earlier software versions.

```
access-list 1 permit 172.30.0.0 0.0.255.255
access-list 1 permit 172.30.0.0 0.0.255.255
!
line 1 40
 access-class 1 out
! Define LAT access list that permits all connections.
 lat access-list 1 permit .*
```

LAT Connection Examples

The following example establishes a LAT connection from the router named router to host eng2:

```
Router> lat eng2
Trying ENG2...Open
         ENG2 - VAX/VMS V5.2
Username: JSmith
Password: <password>
    Welcome to VAX/VMS version V5.2 on node ENG2
    Last interactive login on Friday,  1-APR-1994 19:46
```

The system informs you of its progress by displaying the messages “Trying <system>...” and then “Open.” If the connection attempt is not successful, you receive a failure message.

The following example establishes a LAT connection from the router named router to our-modems and specifies port 24, which is a special modem:

```
Router> lat our-modems port 24
```

The following example establishes a LAT connection from the router named router to our-modems and specifies a node named eng:

```
Router> lat our-modems node eng
```

The following example uses the LAT session debugging capability:

```
Router> lat Eng2 /debug
Trying ENG2...Open
        ENG2 - VAX/VMS V5.2
Username: JSmith
Password: <password>
    Welcome to VAX/VMS version V5.2 on node ENG2
    Last interactive login on Tuesday, 5-APR-1994 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 ([ ]) are the messages produced by the remote system setting the line characteristics as the operating system defaults.

The following example defines a group code list for the outgoing group 4 LAT connection:

```
Router> terminal lat out-group 4, 6-189
```

Configuring TN3270

IBM 3270 display terminals are among the most widely implemented and emulated terminals for host-based computing in the computing community. Information in this section describes the TN3270 terminal emulation environment and how to use and create files that allow terminals connected to the access server or router to be used for TN3270 operation.

This section does not describe how to configure a TN3270 server. For information about configuring TN3270 server support in the Cisco IOS software, see the *Cisco IOS Bridging and IBM Networking Configuration Guide.*

The following sections are included:

TN3270 Overview

TN3270 terminal emulation software allows any terminal to be used as an IBM 3270-type terminal. Users with non-3270 terminals can take advantage of the emulation capabilities to perform the functions of an IBM 3270-type terminal. The Cisco IOS software supports emulation of the following terminal types:

- IBM 3278-2 terminal with an 80-by-24 display
- IBM 3278-2 terminal with a 24-by-80 display
- IBM 3278-3 terminal with a 32-by-80 display
- IBM 3278-4 terminal with a 48-by-80 display
- IBM 3278-5 terminal with a 27-by-132 display

True IBM 3270-type terminals use a character format referred to as Extended Binary Coded Decimal Interchange Code (EBCDIC). EBCDIC consists of 8-bit coded characters and was originally developed by IBM. Emulation is made possible by the termcap protocol. Termcap functions translate the keyboard and terminal characteristics for ASCII-type terminals into those required for an IBM host.

Formally, a termcap is a two-part terminal-handling mechanism. It consists of a database and a subroutine library. The database describes the capabilities of each supported terminal, and the subroutine library allows programs to query the database and to make use of the values it contains. For more information about defining termcaps, refer to the commercially available book *termcap & terminfo*, by Jim Strang, Tim O'Reilly, and Linda Mui.

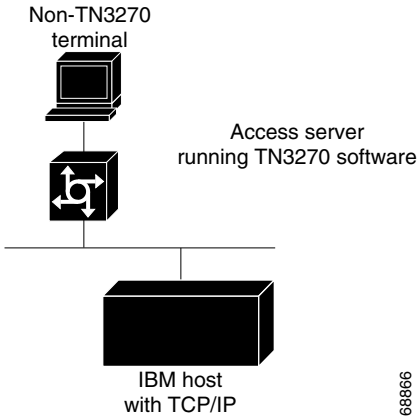
The Cisco IOS software includes a default termcap entry for Digital VT100 terminal emulation. More samples are available directly from Cisco at http://www.cisco.com/warp/public/494/1.html. This URL is subject to change without notice.

TN3270 emulation capability allows users to access an IBM host without using a special IBM server or a UNIX host acting as a server. (See Figure 2.) The IBM host must directly support TCP/IP or have a front-end processor that supports TCP/IP.

A two-step translation method connects IBM hosts from LAT, TCP, and X.25/PAD environments. (See the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" later in this publication for more information about two-step translations.) In general, TN3270 support allows

outgoing TN3270 connections only. In other words, LAT, TCP, and X.25/PAD users must first establish a connection with the access server or router, then use the TN3270 facility from the Cisco IOS software to make a connection to the IBM host.

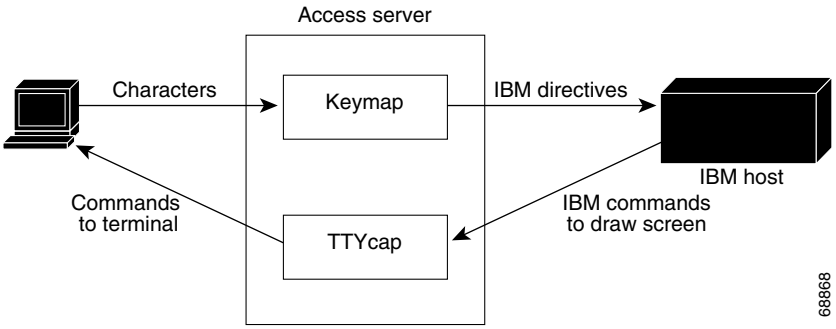
Figure 2 Typical TN3270 Connection Environment



Keymaps and ttycaps

Figure 3 shows how the keymapping and TTYcap functionality in the Cisco IOS software allows IBM hosts and non-IBM terminals to communicate.

Figure 3 Keymaps and TTYcaps



Keymaps and TTYcaps have the following functionality:

- Keymap—Keyboard map file. Terminals send a key sequence for every key used to send packets to an IBM host. The keymapping function in the Cisco IOS software identifies special sequences and converts them to directives to the IBM host. A minimal level of keymapping is supported by default. Several keys can convert to the same IBM directives.
- TTYcap—Terminal emulation file. IBM devices and software send commands to the terminal, including cursor position, clear screen, and so on. The TTYcap functionality in the Cisco IOS software changes IBM directives into the terminal language. By default, protocol translation on access servers and routers conforms to the American National Standards Institute (ANSI) terminal standard, which is VT*xxx* terminal compatible.

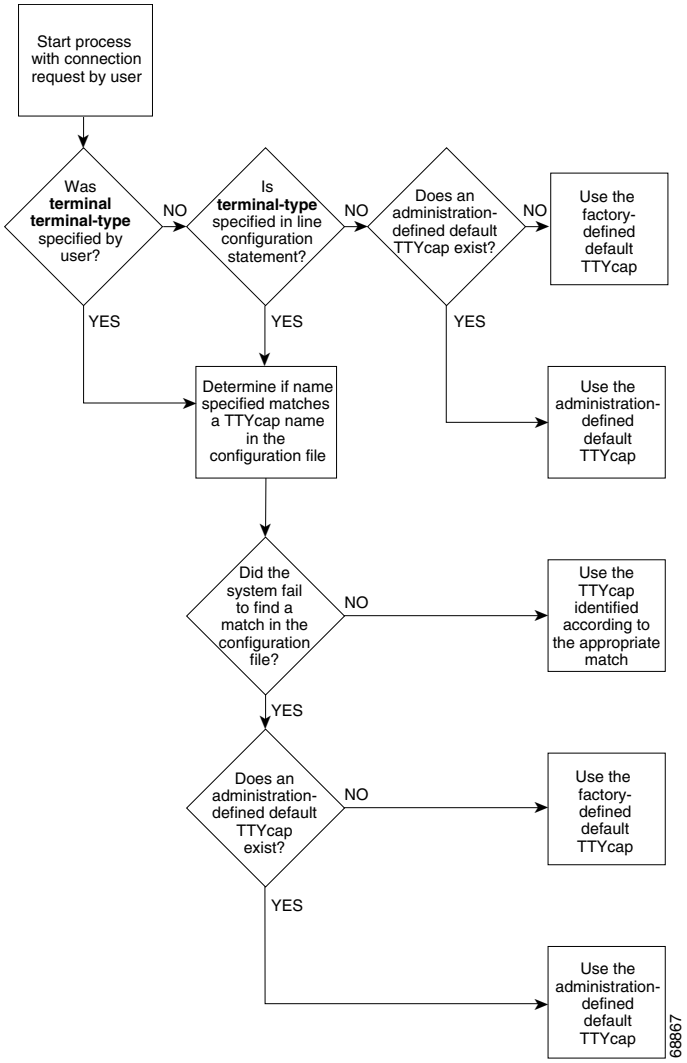
Startup Sequence Priorities

At system startup, the Cisco IOS software uses the following decision sequence when selecting a TTYcap:

1. Use a user-supplied terminal emulation filename.
2. Use a terminal emulation filename specified using line configuration commands.
3. Use a default terminal emulation filename supplied by the administrator.
4. Use the default VT100 emulation.

Figure 4 illustrates the decision process used by the Cisco IOS software to choose a TTYcap for a specific TN3270 session.

Figure 4 Decision Diagram for Cisco IOS Software TTYcap Selection Process



At system startup, the Cisco IOS software uses the following decision sequence when selecting a keymap:

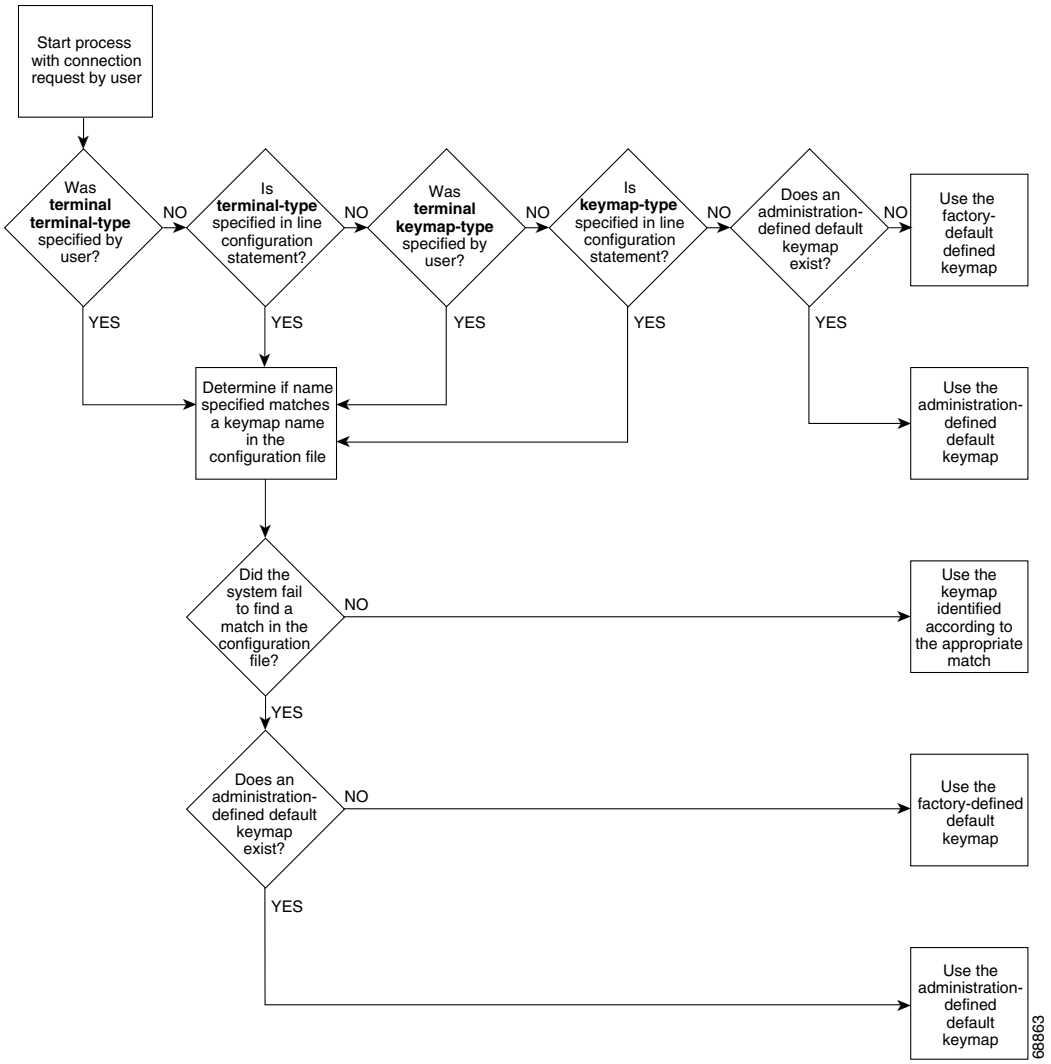
1. Use a user-supplied keyboard map filename.
2. Use a keyboard map filename specified using line configuration commands.
3. Use a user-supplied terminal emulation filename.
4. Use a terminal emulation filename specified using line configuration commands.
5. Use the default keyboard map filename supplied by the administrator.
6. Use the default VT100 emulation.

The software uses the following criteria to determine the file to use:

- If a filename is specified by the user but fails to match any name in the configuration file, the access server or router adopts the default specified by the administrator. If one has not been specifically defined, the factory-default emulation file is adopted.
- If a filename is specified for line configuration that does not match any name in the configuration file, the access server or router adopts the default specified by the administrator. If one has not been specifically defined, the factory-default VT100 emulation file is used.

Figure 5 illustrates the decision process used by the Cisco IOS software to choose a keymap for a specific TN3270 session. When one of the first four priority checks fails (that is, the name specified does not match any name in the configuration file), the same rules listed for the terminal emulation file apply.

Figure 5 Decision Diagram for Cisco IOS Software Keymap Selection Process



Using the Default Terminal Emulation File to Connect

By default, an ASCII terminal and keyboard connected to the Cisco device emulate a Digital VT100 terminal type.

To connect to an IBM host, enter the **tn3270** command from EXEC mode. This command will make the connection using the terminal emulation file selected using the startup sequence priorities outlined in "Startup Sequence Priorities" earlier in this section.

Refer to the "Configuring TN3270 Connections" section later in this document for more information about making connections.

Copying a Sample Terminal Emulation File

If the default file does not work for your terminal and keyboard type or the host that you connect to, you might be able to find a usable file from the growing list of sample terminal emulation files created by Cisco engineers and customers. You can obtain the TN3270 examples from Cisco.com. Numerous emulation files are listed in the examples, which allow various terminal types to emulate an IBM 3270-type terminal.

To obtain these sample configuration files, perform the following steps:

Step 1 Obtain a sample configuration file from the following URL. The *TN3270 Keymap Examples* document appears. Note that this URL is subject to change without notice.

```
http://www.cisco.com/warp/public/494/1.html

TN3270 Keymap Examples
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! TN3270 examples file
! For use with the TN3270 on the cisco terminal server
! If you have requests for additions, contact tac@cisco.com
! If you have contributions, send them to remaker@cisco.com
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! Example of a ttycap for a televideo 925
! Taken from standard TTYCAP from BSD Unix
!
ttycap televideo \
v8|vi|tvi925|925|televideo model 925:\
        :hs:am:bs:co#80:li#24:cm=\E=%+ %+ :cl=\E*:cd=\Ey:ce=\Et:\

:al=\EE:dl=\ER:im=:ei=:ic=\EQ:dc=\EW:mr=\EG4:mk=\EG1:md=\EG4:me=\EG0:\
        :ho=^^:nd=^L:bt=\EI:pt:so=\EG4:se=\EG0:sg#1:us=\EG8:ue=\EG0:ug#1:\
        :up=^K:do=^V:kb=^H:ku=^K:kd=^V:kl=^H:kr=^L:kh=^^:ma=^V^J^L :\
        :k1=^A@\r:k2=^AA\r:k3=^AB\r:k4=^AC\r:k5=^AD\r:k6=^AE\r:k7=^AF\r:\
        :k8=^AG\r:k9=^AH\r:k0=^AI\r:ko=ic,dc,al,dl,cl,ce,cd,bt:\
        :ts=\Ef:fs=\Eg:ds=\Eh:sr=\Ej:xn:ti=\EG0:to=\EG0:\
        :is=\El\E"^M\E3^M        \E1        \E1        \E1        \E1
\E1        \E1        \E1        \E1        \E1^M
!
! Example of a keymap for a 925
! Borrowed from MAP3270 of the BSD TN3270
!
...
```

Step 2 Use a text editor or word processing application to copy the sample terminal emulation file into the configuration file.

Step 3 Load the configuration file onto the host or network. (Refer to the chapter "Loading System Images and Configuration Files" in the *Cisco IOS Configuration Fundamentals Configuration Guide*, for information on loading configuration files.)

This procedure adds new terminal emulation capability to the configuration file. Each time the system is started up, or booted, the settings in the file will be used as the default for terminal emulation.

TN3270 Configuration Task List

To configure TN3270, perform the tasks in the following sections:

- Configuring TN3270 Connections (Required for Service)
- Mapping TN3270 Characters (As Required)
- Starting TN3270 Sessions (Required for Making Connections)

The section "TN3270 Configuration and Connection Examples" later in this chapter provides examples of making TN3270 connections.

Configuring TN3270 Connections

The tasks in this section indicate how to create TTYcap and keymap files, and configure your lines for a TN3270 connection.

To create a TTYcap and keymap file, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ttycap ttycap-name termcap-entry`	Creates a custom terminal emulation file, or TTYcap.
Step 2	`Router(config)# keymap keymap-name keymap-entry`	Creates a custom keyboard emulation file, or keymap.

To configure your line for the TN3270 connection, use the following commands in line configuration mode:

	Command	Purpose
Step 1	`Router(config-line)# terminal-type terminal-name`	Specifies the type of terminal connected to the line.
Step 2	`Router(config-line)# keymap-type keymap-name`	Specifies the keyboard map for a terminal connected to the line.

To customize the TN3270 connection environment, use the following commands in global configuration mode. (These tasks are optional).

	Command	Purpose
Step 3	`Router(config)# tn3270 datastream {extended | normal}`	Enables TN3270 extended features.
Step 4	`Router(config)# tn3270 null-processing [3270 | 7171]`	Enables null processing.
Step 5	`Router(config)# tn3270 reset-required`	Specifies a reset whenever a 3278-x terminal keyboard locks up.

To use a custom emulation file, you must load the emulation settings into the system configuration file. This step establishes the settings in the file as the terminal and keyboard defaults and provides several ways in which the emulation settings can be used within the system, as follows:

- You can provide default settings for all terminals in the network or terminals on a specific host.

- You can set up your system to boot, or load, a specific configuration file using configuration commands described in the *Cisco IOS Configuration Fundamentals Configuration Guide*, Release 12.2.
- You can temporarily override default settings using terminal EXEC commands.
- Load in the files by using the local **terminal terminal-type** and **terminal keyboard-type** EXEC command*s*.
- You can configure line-specific emulation types for terminal negotiations with a remote host.

If you intend to use an alternate TTYcap and keymap, you must assign the following two characteristics:

- Terminal type
- Keymap type

The terminal and keymap type information is used by the Cisco IOS software when negotiating connections with hosts. Use the **terminal-type** and **keymap-type** line configuration commands to assign TTYcap and keymap line characters. You must assign the terminal and keyboard type to the line if you intend to use alternate TTYcap and keymap files.

Use the **tn3270 datastream** command to cause an "-E" to be appended to the terminal type string sent to the IBM host. This command allows you to use the extended TN3270 features.

If a user enters data, uses an arrow key to move the cursor to the right on the screen, and then enters more data, the intervening spaces are filled in with nulls. To specify how nulls are handled, enter the **tn3270 null-processing** command either with the argument **3270**, where nulls are compressed out of the string (as on a real 3278-x terminal), or use the **7171** argument, where nulls are converted to spaces as on a 7171 controller.

On a 3278-x terminal, the keyboard is locked and further input is not permitted after an input error (due to field overflow, invalid entry, and so on), until the user presses the RESET key. Most TN3270 implementations leave the keyboard unlocked and remove any error message on the next key input after the error. Use the **tn3270 reset-required** command to enable a reset in these situations.

Mapping TN3270 Characters

To control the mapping of EBCDIC and ASCII characters, use the following commands in the modes indicated, as needed:

Command	Purpose
`Router(config)# tn3270 character-map` *ebcdic-in-hex ascii-in-hex*	In global configuration mode, creates character mappings by configuring a two-way binding between EBCDIC and ASCII characters.
`Router(config)# no tn3270 character-map` {**all** | *ebcdic-in-hex*} [*ascii-in-hex*]	In global configuration mode, resets character mappings to their default settings.
`Router> show tn3270 character-map` {**all** | *ebcdic-in-hex*}	In EXEC mode, displays character mappings.
`Router> show tn3270 ascii-hexval`	In EXEC mode, displays the hexadecimal value of an ASCII character.[1]

Command	Purpose
`Router(config-line)# tn3270 8bit display`	In line configuration mode, temporarily configures the Cisco IOS software to use the 8-bit mask.
`Router(config-line)# tn3270 8bit transparent-mode`	In line configuration mode, temporarily configures the Cisco IOS software to use the 8-bit mask if you use a file-transfer protocol such as Kermit in 8-bit mode.

1. After you enter the **show tn3270 ascii-hexval** command, enter the ASCII character whose hexadecimal value you want to display.

When you create character mappings between extended EBCDIC or extended ASCII characters, you must configure the Cisco IOS software for the correct data character bit length. The default mask used for TN3270 connections is a 7-bit mask. In certain situations, you must use an 8-bit display. When an 8-bit mask has been set by the **data-character-bits** {**7** | **8**} line configuration command or the **terminal data-character-bits** {**7** | **8**} EXEC command, you can temporarily configure the software to use the 8-bit mask by entering the **tn3270 8bit display** line configuration command.

When you use a file-transfer protocol such as Kermit in 8-bit mode or you use 8-bit graphics, which rely on transparent mode, use the **tn3270 8bit transparent-mode** line configuration command to configure the software for the 8-bit mask.

Starting TN3270 Sessions

You use TN3270 terminal emulation to connect to an IBM 3278-type host. Your system administrator must configure a default terminal emulation file that permits the terminal to communicate with the host. How to specify alternate terminal emulations is described in the section "Configuring TN3270 Connections" earlier in this chapter.

Unlike with Telnet and LAT connections, you *must* enter the **tn3270** command to make a connection to an IBM 3278-type host. To start a TN3270 session, use the following command in EXEC mode:

Command	Purpose
`Router> tn3270 host [keyword]`	Begins a TN3270 session. Refer to the description of the **tn3270** command in the *Cisco IOS Terminal Services Command Reference*, for a list of supported keywords.

To terminate an active TN3270 session, enter the escape sequence (Ctrl-Shift-6 then x [Ctrl^x] by default) and enter the **disconnect** command at the EXEC prompt. You can also log out of the remote system by issuing the command specific to that system (such as **exit**, **logout**, **quit**, **close**, or **disconnect**). For an example of making TN3270 connections, see the next section, "TN3270 Configuration and Connection Examples."

TN3270 Configuration and Connection Examples

This section provides the following examples to help you define custom terminal and keyboard emulation files, and to configure your system to use those files:

- Custom Terminal Emulation File Example
- Custom Keyboard Emulation File Example

Custom Terminal Emulation File Example

The following example allows a Televideo 925 terminal to emulate an IBM 3270-type terminal. The file is part of the global **ttycap** command and is included in the system configuration file. Notice that a carriage return (^M) indicates the last character in the file.

```
ttycap ttycap1 \
v8 | vi | tvi925 | 925 | televideo model 925:\
        :so=\EG4:se=\EG0:\
        :hs:am:bs:co#80:li#24:cm=\E=%+ %+ :cl=\E*:cd=\Ey:ce=\Et:\
        :al=\EE:dl=\ER:im=:ei=:ic=\EQ:dc=\EW:\
        :ho=^^:nd=^L:bt=\EI:pt:so=\EG4:se=\EG0:sg#1:us=\EG8:ue=\EG0:ug#1:\
        :up=^K:do=^V:kb=^H:ku=^K:kd=^V:kl=^H:kr=^L:kh=^^:ma=^V^J^L :\
        :k1=^A@\r:k2=^AA\r:k3=^AB\r:k4=^AC\r:k5=^AD\r:k6=^AE\r:k7=^AF\r:\
        :k8=^AG\r:k9=^AH\r:k0=^AI\r:ko=ic,dc,al,dl,cl,ce,cd,bt:\
        :md=\E(:me=\E):ti=\E):te=\E(:\
        :ts=\Ef:fs=\Eg:ds=\Eh:sr=\Ej:xn:\
        :is=\El\E"^M\E3^M      \E1        \E1        \E1       \E1       \E\
1        \E1        \E1        \E1        \E1^M
```

Custom Keyboard Emulation File Example

The following example allows a keyboard to emulate an asynchronous connection to an IBM 7171 keyboard. The file is part of the **keymap** global configuration command and is included in the system configuration file.

```
keymap ibm7171 \
vt100av | vt100 | vt100nam | pt100 | vt102 | vt125{ \
enter = '^m';\
erase = '^?'; reset = '^g'; clear = '^z'  |  '\EOM';\
nl = '^j'; tab = '^i'; btab = '^b';\
left = '\EOD'; right = '\EOC'; up = '\EOA'; down = '\EOB';\
home = '^h'; delete = '^d'; eeof = '^e'  |  '\E^?'; einp = '^w'; insrt = '\EOn';\
pfk1 = '\EOP'  |  '\E1'; pfk2 = '\EOQ'  |  '\E2'; pfk3 = '\EOR'  |  '\E3';\
pfk4 = '\EOw'  |  '\E4'; pfk5 = '\EOx'  |  '\E5'; pfk6 = '\EOy'  |  '\E6';\
pfk7 = '\EOt'  |  '\E7'; pfk8 = '\EOu'  |  '\E8'; pfk9 = '\EOv'  |  '\E9';\
pfk10 = '\EOq'  |  '\E0'; pfk11 = '\EOr'  |  '\E-';\
pfk12 = '\EOs'  |  '\E='; pfk13 = '\EOp\EOP'  |  '^f13';\
pfk14 = '\EOp\EOQ'  |  '^f14'; pfk15 = '\EOp\EOR'  |  '^f15';\
pfk16 = '\EOp\EOw'  |  '^f16'; pfk17 = '\EOp\EOx'  |  '^f17';\
pfk18 = '\EOp\EOy'  |  '^f18'; pfk19 = '\EOp\EOt'  |  '^f19';\
pfk20 = '\EOp\EOu'  |  '^f20'; pfk21 = '\EOp\EOv'  |  '^f21';\
pfk22 = '\EOp\EOq'  |  '^f22'; pfk23 = '\EOp\EOr'  |  '^f23';\
pfk24 = '\EOp\EOs'  |  '^f24';\
pa1 = '^p1'  |  '\EOS';\
pa2 = '^p2'  |  '\EOm';\
pa3 = '^p3'  |  '\EOl';\
}
```

Line Specification for a Custom Emulation Example

The following example sets up a line with specific terminal and keyboard characteristics that are used during negotiation with a host upon connection. The line configuration commands in the example must follow the global **ttycap** and **keymap** global configuration commands containing the emulation settings to be used.

```
line 3
 terminal-type ttycap1
 keymap-type ibm7171
```

Character Mapping Examples

The following example shows the configuration of the EBCDIC and ASCII character mappings listed in Table 1:

```
tn3270 character-map 0x81 0x78
tn3270 character-map 0x82 0x79
tn3270 character-map 0x83 0x7A
```

Table 1 Sample EBCDIC and ASCII Character Mapping

EBCDIC	ASCII
a	x
b	y
c	z

The following example displays all nonstandard character mappings:

```
Router# show tn3270 character-map all

EBCDIC 0x81 <=> 0x78 ASCII
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
```

The following example shows the standard key mapping for the letters d and c:

```
Router# show tn3270 character-map 83

EBCDIC 0x83 <=> 0x63 ASCII = `c'
EBCDIC 0x84 <=> 0x64 ASCII = `d'
```

The following example unmaps a specific key, first with the optional *ascii-in-hex* argument and then without the argument:

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# no tn3270 character-map 0x80 0x78
Router(config)# ^Z

Router# show tn3270 character-map all

EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII

Router# configure terminal
```

```
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# no tn3270 character-map 0x82
Router(config)# ^Z
Router# show t3270 character-map all

EBCDIC 0x82 <=> 0x79 ASCII
```

The following example displays character mappings, then removes all mappings with the **all** keyword:

```
Router# show tn3270 character-map all

EBCDIC 0x81 <=> 0x78 ASCII
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII

Router# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# no tn3270 character-map all
Router(config)# ^Z

Router# show tn3270 character-map all
```

TN3270 Connection Example

The following example establishes a terminal session with an IBM TN3270 host named finance and specifies vt100 as the terminal type:

```
Router> tn3270 finance /terminal-type vt100
```

To terminate an active TN3270 session, log out of the remote system by entering the command specific to that system (such as **exit**, **logout**, **quit**, or **close**). You can also enter the escape sequence (**Ctrl-Shift-6** then **x** [**Ctrl^x**] by default) and enter the **disconnect** command at the EXEC prompt. Because the **disconnect** command can "hang" a port, we recommend that you avoid using it routinely when you exit a session.

TN3270 Menu Example

The following example shows the use of the **/terminal-type** *type* keyword and argument combination when using tn3270 with menus:

```
menu router1 text 1 Connect from client
    menu router1 command 1 tn3270 router1.com /term h19
    menu router1 text 2 Connect from VT-100
    menu router1 command 2 tn3270 router1.com /term vt100
    menu router1 text 3 Connect from PC running Procomm
    menu router1 tn3270 router1.com /term vt100-pc
```

Configuring XRemote

The X Window System, also called X, is a network-based graphics window system originally developed for workstations running UNIX. Cisco has developed an XRemote application that allows the XRemote capabilities of X terminals to run on an access server or router.

Previous window systems for terminals were *kernel-based* and therefore were closely linked to the operating system running on the workstation itself. They typically only ran on discrete systems, such as a single workstation. The X Window System is not part of any operating system, but instead, is composed of application programs. Thus, the X Window System enables flexible, graphics-based network computing across a wide range of operating systems and hardware platforms.

X and the Client/Server Model

The underlying architecture of the X Window System is based on a *client/server* model. The system is split into two parts: *clients* and *display servers*. Clients are application programs that perform specific tasks, and display servers provide specific display capabilities and track user input. These two parts can reside on the same computer or can be separated over a network. In an X terminal environment, such as in NCD terminal implementations, the display server resides on the display station and the client resides on a host computer.

Because the X Windows System employs this client/server partitioning and is independent of both the hardware and operating environment, X terminal users can access different types of computers to simultaneously access several applications and resources in a multivendor environment. A user at an X terminal can concurrently run and display a calendar program on a VAX, a spreadsheet program on a PC, and a compiler on a workstation.

XRemote Overview

Note Beginning with Cisco IOS XE Gibraltar 16.11.1, this protocol is no longer supported.

XRemote is a protocol developed specifically to optimize support for the X Window System over a serial communications link. Its compression and decompression algorithms are designed to handle bit-mapped displays and windowing systems.

There are two basic parts to XRemote:

- Server-side helper process
- Client-side helper process

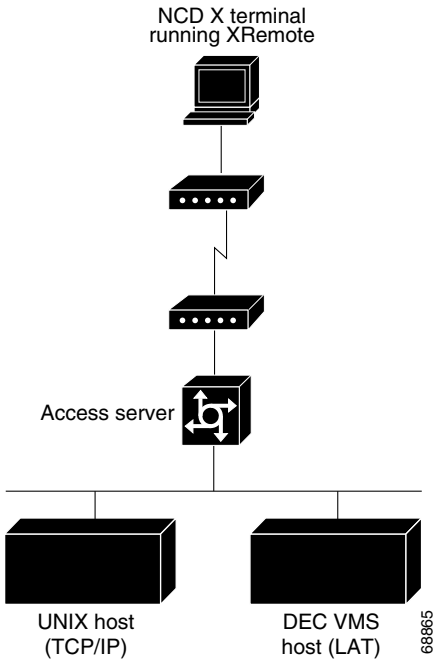
These two helper processes communicate with each other using the XRemote protocol. The client-side helper communicates with X clients using the standard X protocol. The server-side helper communicates with the server using the standard X Window System. The server-side helper might operate as part of the X server or it might be external and accessed across the network; for example, the server-side helper can operate in an access server or router at your house or work site. If the server-side helper is in the X terminal, it must have XRemote programmable read-only memory (PROM) installed.

XRemote enables a user of a display station to run the X Window System via 9600-baud (and faster) modem connections with performance that is superior to using conventional serial protocols, such as Serial Line Internet Protocol (SLIP). An X display station must either implement XRemote or be connected to a network configuration that includes an access server or router.

Connection Capability

The Cisco implementation of XRemote is fully compatible with the NCD XRemote protocol. Figure 6 illustrates an XRemote connection between an X terminal and an access server. In Figure 6, the server-side helper runs on the X terminal, and the client-side helper runs on the access server.

Figure 6 XRemote Session from an X Display Server Running XRemote



Remote Access to Fonts

Remote access to fonts is provided in three ways:

- Using the industry-standard protocol for transporting X traffic over TCP/IP networks
- Using the Digital protocol for transporting X traffic over LAT networks
- Using the Internet standard TFTP for TCP/IP networks

A single XRemote user can use any combination of TCP/IP and LAT client connections and any combination of TFTP and LAT font access.

XRemote Configuration Task List

To configure XRemote, perform the tasks described in the following sections:

- Configuring XRemote (Required for Service)
- Selecting Fonts for X Terminal Applications (Optional)
- Making XRemote Connections (Required for Making Connections)

The section "Monitoring XRemote Connections" provides tips on maintaining XRemote connections.

Configuring XRemote

To allow host connections using the XRemote feature from NCD and the access server or router, use the following commands. Before starting the following tasks, verify that a modem is externally or internally connected with your access server or router. Unless specified otherwise, all commands in this task table are entered in global configuration mode.

	Command	Purpose[1]
Step 1	`Router(config)# xremote tftp host hostname`	Defines a specific TFTP font server as the source for fonts.
Step 2	`Router(config)# xremote tftp buffersize buffersize`	Sets the buffer size used for loading font files.
Step 3	`Router(config)# xremote tftp retries retries`	Increases the number of times that the font loader tries to load the fonts.[2]
Step 4	`Router> show xremote`	(Optional) In EXEC mode, displays current XRemote connections and monitors traffic.
Step 5	`Router> show xremote line number`	(Optional) In EXEC mode, displays XRemote traffic and line statistics.

1. The X Server for the X terminal and the network and serial parameters for the X terminal must be configured as described in the publications for the specific X terminal you are using. In general, the X terminal configuration determines the mode of operation for the terminal, the source of font information, and the source of remote configuration information (when applicable).
2. This feature is particularly useful when the font servers are known to be heavily loaded.

In general, you can use any modem that provides acceptable performance for your application. The following guidelines apply to an XRemote operation using a modem (see the user manual for your modem for specific connection procedures):

- Attach cables and set up your modem for use with XRemote (access over asynchronous lines only), or cable the X terminal directly to the access server or router.
- Disable any error correction and compression features of the modem. Because XRemote implements its own compression and error correction, the compression and error correction from the modem actually impair performance.
- If you must use a flow control mechanism, hardware flow control (such as RTS/CTS or DTR/DSR) is recommended. Software flow control (such as XON/XOFF) is discouraged.
- The modem should incur minimal delays in round-trip transmissions, even when transmitting small packets, and transmissions should be transparent to the data stream.
- The modem should provide true full-duplex transmission at 9600 baud or faster. Half-duplex modems are not suitable for use with XRemote.

Refer to *Cisco IOS Dial Technologies Configuration Guide*, for more information about configuring modems.

When the X terminal requests that a font file be loaded, the Cisco IOS software must first load the font file into an internal buffer before passing it to the X terminal. The default value for this buffer is 70000 bytes, which is adequate for most font files, but the size can be increased as necessary for nonstandard font files using the **xremote tftp buffersize** global configuration command. This task can be performed for both TFTP and LAT font access.

Selecting Fonts for X Terminal Applications

The NCD terminal contains a small set of built-in fonts in local ROM. You should use these fonts because loading fonts over a serial line can increase application startup time. The default for an NCD terminal is to use built-in fonts, unless you log in using DECwindows over LAT. When using DECwindows over LAT, the standard DECwindows fonts are used automatically.

To select fonts, perform the tasks described in the following sections:

- Accessing Nonresident Fonts Using TFTP
- Selecting DECwindows Fonts

Accessing Nonresident Fonts Using TFTP

When an X terminal application requests a font that is not stored in ROM for the terminal, the X terminal makes a request for a font file from the access server or router. The Cisco IOS software uses the TFTP to load the font from the font server, and then passes the font to the X terminal using the XRemote protocol. Loading fonts from the access server or router to the X terminal can take 30 to 45 seconds, depending on the size of the font file.

An X server can display only the fonts it finds in the directories in its font path. The default font path for the X server includes only the built-in fonts. To access fonts stored on a host, you must add the font directories from the host to the font path of the X server, which is done using the UNIX command **xset** with the **fp+** argument to add fonts to the end of the font path of the server.

For example, to allow your display station to access the 100 dots-per-inch (dpi) fonts found in the standard font directory, enter the following command at the host system prompt:

```
host_prompt% xset fp+ /usr/lib/x11/ncd/fonts/100dpi
```

For more information, see the *NCDware XRemote User's Manual*.

Selecting DECwindows Fonts

Downloading of fonts occurs automatically when you initiate a remote DECwindows login session using the **xremote lat** EXEC command. Using the **xremote lat** EXEC command instead of relying on TFTP to download the fonts, the fonts are read in via the LAT protocol.

If you want to use DECwindows fonts while running standard X applications on a UNIX host, you need to use the UNIX **xset** command or an application that sends an XSetFontPath request to set a font path. You might want to use the UNIX **xset** command if you are primarily a TCP/IP user, but also run some DECwindows applications.

Enter the **xset** command, or launch the application that sends an XSetFontPath request, to set the following path:

```
/LAT/SERVICE
```

In this path, SERVICE is a LAT service name with DECwindows support; case is not significant.

When the Cisco IOS software sees a request for font files in that directory, it uses LAT instead of TFTP to access the specified service.

Making XRemote Connections

You use the XRemote protocol with an X display station and a modem to connect to remote hosts via TCP/IP and LAT. This section outlines the steps for starting XRemote in several typical environments and for exiting XRemote sessions. It includes the following sections:

- Connecting Through Automatic Session Startup with an XDMCP Server
- Connecting Through Automatic Session Startup with a DECwindows Login via LAT
- Connecting Through Manual XRemote Session Startup
- Establishing XRemote Sessions Between Servers
- Exiting XRemote Sessions

When possible, use the automated processes. Make sure that your system administrator has already configured a path for loading fonts.

You can run the XRemote protocols between two servers. This capability is useful if you use an X display server that does not support XRemote, or if an X display station is connected to a LAN and you want to use the LAN rather than a dial-in link to connect to a server. (Note that XRemote is faster when the X display station connects to a server over a dial-in link.) Refer to the section "Establishing XRemote Sessions Between Servers" later in this chapter.

For an example of making an XRemote connection, see the "XRemote Configuration and Connection Examples" section later in this chapter.

Connecting Through Automatic Session Startup with an XDMCP Server

If your host computer supports a server for X Display Manager Control Protocol (XDMCP) (such as the xdm program included in X11R4 or later), you can use automatic session startup to make an XRemote session connection. To do so, use the following command in EXEC mode:

Command	Purpose
`Router> xremote xdm [hostname]`	Creates a connection with XRemote and an XDMCP server.

This command sends an XDMCP session startup request to the host computer. If you do not specify a host name, a broadcast message is sent to all hosts. The first host to respond by starting up a session is used.

The server 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.

Connecting Through Automatic Session Startup with a DECwindows Login via LAT

If your host computer supports DECwindows login sessions, you can use automatic session startup to make an XRemote session connection, when the system administrator at the remote host configures support for DECwindows over LAT. To start the connection, use the following command in EXEC mode:

Command	Purpose
`Router> xremote lat service`	Creates a connection with XRemote and DECwindows over LAT.

After you enter this command, expect the following to occur:

- The XRemote font server loads several initial fonts for the DECwindows login display.
- The terminal displays the Digital logo and DECwindows login box.

Log in to the system. Upon completion of login, more fonts are loaded, and the remote session begins.

Note Because of heavy font usage, DECwindows applications can take longer than expected to start when you use XRemote. After the application starts, performance and access times should be normal.

Connecting Through Manual XRemote Session Startup

If you do not use a host computer that supports XDMCP or LAT, you must use manual session startup. To use manual session startup, perform the tasks described in the following sections:

- Enabling XRemote Manually (Required for Manual Sessions)
- Connecting to the Remote Host Computer (Required for Manual Sessions)
- Setting the Location of the X Display (Required for Manual Sessions)
- Starting Client Applications (Required for Manual Sessions)
- Returning to the EXEC Prompt (Required for Manual Sessions)
- Reenabling XRemote Manually (Required for Manual Sessions)

Enabling XRemote Manually

To prepare the XRemote server for manual startup, use the following command in EXEC mode:

Command	Purpose
`Router> xremote`	Prepares the XRemote server for manual startup.

After you enter this command, instructions prompt you through the process of manually enabling XRemote.

Note In manual operation, the server and X terminal remain in XRemote mode until all clients disconnect or the server receives a reset request from the X terminal. A session might terminate during startup because you invoked transient X clients that set some parameters and then disconnected (such as **xset** or **xmodmap** parameters). There must always be one session open or the connection is reset.

Connecting to the Remote Host Computer

To connect to a host, use one of the following commands in EXEC mode:

Command	Purpose
`Router>` **`telnet`** or `Router>` **`lat`** or `Router>` **`rlogin`**	Prepares the server for XRemote manual startup.

After entering the command, you can log in as usual.

Setting the Location of the X Display

Note If you are using a version of Telnet on the remote host that supports the "X Display Location" option (RFC 1096), skip this section and go on to the "Starting Client Applications" section.

Once you are logged in to the remote host computer, inform the host computer of your X display location that the server provided when you enabled XRemote manually. For most versions of the UNIX operating system, the X display location is set by using the **setenv** command to set the Display environment variable. Refer to the online X(1) manual page available from UNIX for more information.

On VAX/VMS systems, use the **SET DISPLAY** command to set the X display location. For more information, refer to the *VMS DCL Dictionary*.

Note To set the location of the X display for VAX/VMS client systems, you must install either the TCP/IP transport from Digital or a third-party TCP/IP transport. Contact your VAX/VMS system administrator for the appropriate TCP/IP transport name.

Starting Client Applications

When you ave set the location of the Xdisplay, you can start your client applications for your host operating system, as specified in the documentation for the client applications.

The server accepts the X connection attempt from the client application and places the client in a dormant state.

Returning to the EXEC Prompt

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

If you cannot log out of the host computer and keep your clients running, return to the EXEC prompt for the access server using the escape sequence (Ctrl-Shift-6 then x [Ctrl^x] by default).

Reenabling XRemote Manually

To begin a manual remote session again, see the "Enabling XRemote Manually" section earlier in this chapter. If the X clients connected successfully, the session is put into XRemote mode, and the clients complete their startup.

If no clients are found, you see the following message: "No X clients waiting - check that your display is darkstar:2018"

Check your hosts to determine whether an error has occurred when the session started. The most likely causes are that there is an improperly specified display location, or the host computer did not recognize the name of your server.

Establishing XRemote Sessions Between Servers

If you are on an X display server that does not support XRemote, you can still run the XRemote protocols. An X display server (such as a PCX, MacX, or UNIX workstation) connected to an Ethernet network can dial out through an access server on a conventional modem to access an X client program on a host residing on another network. The access server provides the server-side helper process.

To run XRemote, connect to one of the XRemote ports.

Note The NCD helper process does not support X display devices that use a maximum request and response size larger than 64 kbps.

Find out from your administrator whether the connection from your X display server is configured as an individual line or a rotary connection.

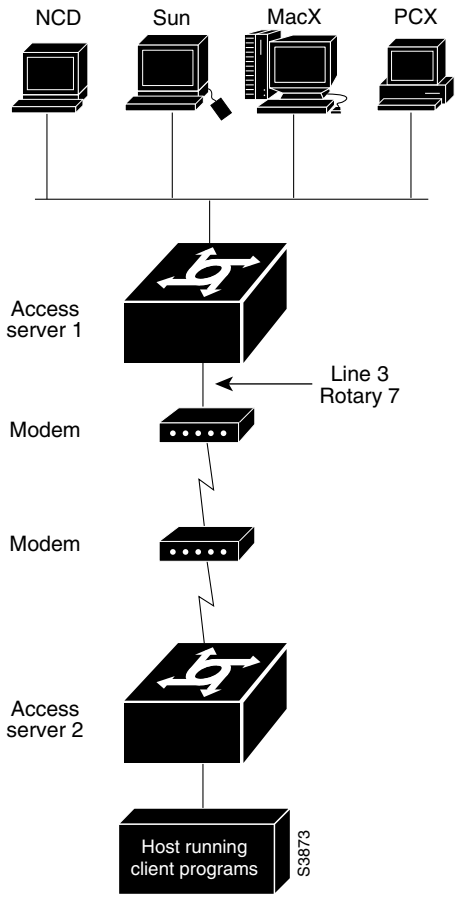
Depending upon the connection configuration, use one of the following connection methods:

- To connect to an individual line, use Telnet to connect from the X display server to port 9000 plus the decimal value of the line number.
- To make a rotary connection, use Telnet to connect from the X display server to port 10000 plus the decimal value of the line number.

For information about how to configure individual lines and rotary connections, refer to *Cisco IOS Dial Technologies Configuration Guide.*

Figure 7 illustrates a configuration in which a display server is not running XRemote. In this configuration, the server-side XRemote helper is running on the access server named Access Server 1, and the client-side XRemote helper is running on the access server named Access Server 2.

Figure 7 XRemote Session Between Servers



Exiting XRemote Sessions

When you exit XRemote, you must quit all active X connections, usually with a command supported by your X client system. Usually when you quit the last connection (all client processes are stopped), XRemote closes and you return to the EXEC prompt. Refer to your X client system documentation for specific information about exiting an XRemote session.

Monitoring XRemote Connections

To list XRemote connections and monitor XRemote traffic through the router, use the following commands in EXEC mode as needed:

Command	Purpose
`Router>` **`show xremote`**	Lists XRemote connections and monitors XRemote traffic through the router or access server.
`Router>` **`show xremote line`** *`number`*	Lists XRemote connections and monitors XRemote traffic for specific lines on an XRemote server.

XRemote Configuration and Connection Examples

These examples are provided to help you understand how to make XRemote connections:

Standard XRemote Configuration Example

The following example shows how to specify IBM-1 as the host name of the TFTP font server, how to specify 7 retry attempts at accessing the server, and how to reduce the buffer size to 20,000 bytes:

```
xremote tftp host IBM-1
xremote tftp retries 7
xremote tftp buffersize 20000
```

Connecting Through Automatic Session Startup with XDMCP Server Example

The following example starts a session with a remote host named star:

```
Router> xremote xdm star
```

Connecting Through Automatic Session Startup with DECwindows Login via LAT Example

The following example begins connection with a LAT service named WHIRL:

```
Router> xremote lat WHIRL
```

Enabling XRemote Manually Example

The following example shows how a successful manual XRemote session begins:

```
Router> xremote
XRemote enabled; your display is dialup:2006
Start your clients and type XRemote again
```

The system replies with a message informing you of your X display location. Use this information to tell the host the location of your X display server.

If no clients are found, you see the following message: “No X clients waiting - check that your display is darkstar:2006”

Check your hosts to determine whether an error has occurred when the session started. The most likely causes are that there is an improperly specified display location or the host computer did not recognize the name of your server.

Connecting an X Display Terminal Example

To make a connection from an X display terminal through a server to a host running client programs, perform the following steps:

Step 1 Enter the **xremote** command at the EXEC prompt:

```
Router> xremote
```

Step 2 Read and follow the instruction from the host:

```
XRemote enabled; your display is dialup:2006
Start your clients and type XRemote again
```

Step 3 Connect to the client:

```
Router> telnet eureka
Trying EUREKA.NOWHERE.COM (172.16.1.55)... Open

SunOS UNIX (eureka)
```

Step 4 Log in at the prompt:

```
login: deal
Password:
Last login: Fri Apr 1 17:17:46 from dialup.nowhere.com
SunOS Release (SERVER+FDDI+DBE.patched) #14: Fri Apr 8 10:37:29 PDT 1994
```

Step 5 At the client prompt, enter the display name from Step 2 in this procedure and the **xterm** command:

```
eureka% setenv DISPLAY dialup:2006
eureka% xterm &
[1] 15439
```

Step 6 Disconnect from the client:

```
eureka% logout

[Connection to EUREKA closed by foreign host]
```

Step 7 Begin the XRemote session:

```
Router> xremote
Entering XRemote
```

The server 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:

```
Connection closed by foreign host.
eureka%
```

Making XRemote Connections Between Servers Example

This section describes two ways to make XRemote connections between servers.

The following process explains how an XRemote connection is established for a configuration such as the one shown in Figure 7 in the section "Establishing XRemote Sessions Between Servers" earlier in this chapter. This procedure assumes that the administrator has set the display environment variable to identify and match the X display terminal of the user.

From the PCX, MacX, or UNIX machine in Figure 7, the user connects to port 9003 on the access server named Access Server 1. If your administrator has configured a rotary number 7, the user connects to port 10007. For more information about rotary groups, refer to *Cisco IOS Dial Technologies Configuration Guide*.

Following is a summary of the connection process:

1. Access Server 1 connects the user to a modem.
2. The modem calls Access Server 2.
3. The user enters the **xremote** command at the Access Server 2 prompt.
4. The user connects to the remote host from Access Server 2 using the **telnet** command.
5. The user starts the X client program that runs on the remote host and displays on the X display server (PCX, MacX, or UNIX host).
6. The user escapes from the remote host back to Access Server 2, or logs out if clients were run in the background, and enters the **xremote** command again at the Access Server 2 prompt.

The following procedure shows a second way to make an XRemote connection between servers. The number 9016 in the first line of the display indicates a connection to individual line 16. If the administrator had configured a rotary connection, the user would enter 10000 plus the number of the rotary (instead of 9016).

Step 1 Enter the **telnet** command to make the connection:

```
space% telnet golden-road 9016
Trying 172.31.7.84 ...
Connected to golden-road.cisco.com.
Escape character is '^]'.
```

Step 2 Supply the password for TACACS verification:

```
User Access Verification

Password: <password>
Password OK

 --- Outbound XRemote service ---
Enter X server name or IP address: innerspace
Enter display number [0]:

Connecting to tty16... please start up XRemote on the remote system
```

Step 3 Dial in to the remote system using the modem, and then log in:

```
atdt 13125554141
DIALING
RING
CONNECT 14400

User Access Verification
Username: deal
Password:
  Welcome to the cisco dial-up access server.
```

Step 4 Enter the **xremote** command at the EXEC prompt, then follow the instructions from the host:

```
Router> xremote
XRemote enabled; your display is dialup:2006
Start your clients and type XRemote again
```

Step 5 Connect to the client:

```
Router> telnet sparks
Trying SPARKS.NOWHERE.COM (173.19.1.55)... Open

SunOS UNIX (sparks)

login: deal
Password: <password>
Last login: Fri Apr 1 17:17:46 from dialup.nowhere.com
SunOS Release (SERVER+FDDI+DBE.patched) #14: Fri Apr 8 10:37:29 PDT 1994
```

Step 6 At the client prompt, enter the display name from Step 4 and the **xterm** command:

```
sparks% setenv DISPLAY dialup:2006
sparks% xterm &
[1] 15439
```

Step 7 Disconnect from the client:

```
sparks% logout

[Connection to SPARKS closed by foreign host]
```

Step 8 Begin the XRemote session.

```
Router> xremote
Entering XRemote
```

When the connection is closed by the foreign host, the Xterm window appears on the local workstation screen:

```
Connection closed by foreign host.
sparks%
```

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

Cisco IOS Software Feature Removal

Feature Overview

The Cisco IOS Software Feature Removal feature is an engineering project to permanently remove selected legacy features (or components) from the IOS code. These features will not be available in future releases of Cisco IOS software.

The legacy features that have been removed as of Release 12.2(13)T are as follows:

- AppleTalk EIGRP
- Apollo Domain
- Banyan VINES
- Exterior Gateway Protocol
- HP Probe
- Interior Gateway Routing Protocol
- Next Hop Resolution Protocol for IPX
- Novell Link-State Protocol
- Simple Multicast Routing Protocol for AppleTalk
- Xerox Network Systems

The legacy features that have been removed as of Release 12.2(15)T are as follows:

- LAN Extension
- Netware Asychronous Services Interface Protocol
- Xremote

This feature module lists the commands that have been removed from or modified in Cisco IOS software with the removal of a specified feature.

Note Commands that have been modified may not all be listed in this document.

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

AppleTalk EIGRP

The following commands have been removed from or modified in Cisco IOS software with the removal of the AppleTalk EIGRP feature. Please note that not all commands that may have been modified are listed here:

- **appletalk eigrp active-time**
- **appletalk eigrp-bandwidth-percentage**
- **appletalk eigrp log-neighbor-changes**
- **appletalk eigrp-splithorizon**
- **appletalk eigrp-timers**
- **appletalk route-redistribution**
- **debug apple eigrp-all**
- **show appletalk eigrp interfaces**
- **show appletalk eigrp neighbors**
- **show appletalk eigrp topology**

Apollo Domain

The following commands have been removed from or modified in Cisco IOS software with the removal of the Apollo Domain feature:

- **apollo access-group**
- **apollo access-list**
- **apollo maximum-paths**
- **apollo network**
- **apollo route**
- **apollo routing**
- **apollo update-time**
- **debug packet**
- **ping**
- **show apollo arp**
- **show apollo interface**
- **show apollo route**
- **show apollo traffic**

Banyan VINES

The following commands have been removed from or modified in Cisco IOS software with the removal of the Banyan VINES feature:

- **clear vines cache**
- **clear vines ipc**

- **clear vines neighbor**
- **clear vines route**
- **clear vines traffic**
- **debug frame-relay**
- **debug packet**
- **debug vines arp**
- **debug vines echo**
- **debug vines ipc**
- **debug vines netrpc**
- **debug vines packet**
- **debug vines routing**
- **debug vines service**
- **debug vines state**
- **debug vines table**
- **show vines access**
- **show vines cache**
- **show vines host**
- **show vines interface**
- **show vines ipc**
- **show vines neighbor**
- **show vines route**
- **show vines service**
- **show vines traffic**
- **trace (VINES)**
- **vines access-group**
- **vines access-list (extended)**
- **vines access-list (simple)**
- **vines access-list (standard)**
- **vines arp-enable**
- **vines decimal**
- **vines encapsulation**
- **vines enhancements**
- **vines host**
- **vines input-network-filter**
- **vines input-router-filter**
- **vines metric**
- **vines neighbor**
- **vines output-network-filter**

- **vines propagate**
- **vines redirect**
- **vines route**
- **vines route-cache**
- **vines routing**
- **vines serverless**
- **vines single-route**
- **vines split-horizon**
- **vines srtp-enabled**
- **vines time access-group**
- **vines time destination**
- **vines time participate**
- **vines time services**
- **vines time set-system**
- **vines time use-system**
- **vines update deltas**
- **vines update interval**

Exterior Gateway Protocol

No commands were removed from or modified in Cisco IOS software with the removal of the EGP feature.

HP Probe

The following commands have been removed from or modified in Cisco IOS software with the removal of the HP Probe feature:

- **arp (interface) probe**
- **ip hp-host**
- **ip probe proxy**

Interior Gateway Routing Protocol

The following commands have been removed from or modified in Cisco IOS software with the removal of the IGRP feature:

- **debug clns igrp packets**
- **debug ip igrp events**
- **debug ip igrp transactions**
- **debug ip routing**

- **default-metric (IGRP)**
- **ip split-horizon (IGRP)**
- **metric holddown**
- **metric maximum-hops**
- **metric weights (IGRP)**
- **neighbor (IGRP)**
- **network (IGRP)**
- **offset-list (IGRP)**
- **router igrp**
- **set metric (IGRP)**
- **timers basic (IGRP)**
- **traffic-share balanced**

LAN Extension

No commands were removed from or modified in Cisco IOS software with the removal of the LAN Extension feature.

Netware Asychronous Services Interface Protocol

The following commands have been removed from or modified in Cisco IOS software with the removal of the NASI protocol:

- **aaa authentication nasi**
- **ipx nasi-server enable**
- **nasi authentication**
- **show ipx nasi connections**

Next Hop Resolution Protocol for IPX

The following commands have been removed from or modified in Cisco IOS software with the removal of the NHRP for IPX feature:

- **clear ipx nhrp**
- **debug nhrp**
- **debug nhrp extension**
- **debug nhrp options**
- **debug nhrp packet**
- **debug nhrp rate**
- **ipx nhrp authentication**
- **ipx nhrp holdtime**

- **ipx nhrp interest**
- **ipx nhrp map**
- **ipx nhrp max-send**
- **ipx nhrp network-id**
- **ipx nhrp nhs**
- **ipx nhrp record**
- **ipx nhrp responder**
- **ipx nhrp use**
- **show ipx nhrp**
- **show ipx nhrp traffic**

Novell Link-State Protocol

The following commands have been removed from or modified in Cisco IOS software with the removal of the NLSP feature:

- **access-list (NLSP)**
- **area-address (NLSP)**
- **clear ipx nlsp neighbors**
- **clear ipx route**
- **clear ipx traffic**
- **deny (NLSP)**
- **distribute-list in**
- **distribute-list out**
- **distribute-sap-list in**
- **distribute-sap-list out**
- **ipx access-list**
- **ipx advertise-default-route-only**
- **ipx flooding-unthrottled**
- **ipx internal-network**
- **ipx nlsp csnp-interval**
- **ipx nlsp enable**
- **ipx nlsp hello-interval**
- **ipx nlsp hello-multiplier**
- **ipx nlsp lsp-interval**
- **ipx nlsp metric**
- **ipx nlsp multicast**
- **ipx nlsp priority**
- **ipx nlsp retransmit-interval**

- **ipx nlsp rip**
- **ipx nlsp sap**
- **ipx ping-default**
- **ipx potential-pseudonode**
- **ipx route**
- **ipx router**
- **log-adjacency-changes**
- **multicast (NLSP)**
- **permit (NLSP)**
- **redistribute (IPX)**
- **route-aggregation**
- **show ipx nlsp database**
- **show ipx nlsp neighbors**
- **show ipx nlsp spf-log**
- **show ipx route**
- **show ipx traffic**

Simple Multicast Routing Protocol for AppleTalk

The following commands have been removed from or modified in Cisco IOS software with the removal of the SMRP for AppleTalk feature:

- **clear smrp mcache**
- **debug smrp all**
- **debug smrp group**
- **debug smrp mcache**
- **debug smrp neighbor**
- **debug smrp port**
- **debug smrp route**
- **debug smrp transaction**
- **show smrp forward**
- **show smrp globals**
- **show smrp group**
- **show smrp mcache**
- **show smrp neighbor**
- **show smrp port**
- **show smrp route**
- **show smrp traffic**
- **smrp mroute-cache protocol appletalk**

- **smrp protocol appletalk**
- **smrp routing**

Xerox Network Systems

The following commands have been removed from or modified in Cisco IOS software with the removal of the XNS feature:

- **access-list (XNS extended)**
- **access-list (XNS standard)**
- **debug xns packet**
- **debug xns routing**
- **show xns cache**
- **show xns interface**
- **show xns route**
- **show xns traffic**
- **xns access-group**
- **xns encapsulation**
- **xns flood broadcast allnets**
- **xns flood broadcast net-zero**
- **xns flood specific allnets**
- **xns forward-protocol**
- **xns hear-rip**
- **xns helper-address**
- **xns input-network-filter**
- **xns maximum-paths**
- **xns network**
- **xns output-network-filter**
- **xns route**
- **xns route-cache**
- **xns router-filter**
- **xns routing**
- **xns ub-emulation**
- **xns update-time**

Xremote

The following commands have been removed from or modified in Cisco IOS software with the removal of the Xremote feature:

- **show xremote**

- **show xremote line**
- **xremote**
- **xremote lat**
- **xremote tftp buffersize**
- **xremote tftp host**
- **xremote tftp retries**
- **xremote xdm**

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2008 Cisco Systems, Inc. All rights reserved.

Configuring AppleTalk Remote Access

This chapter describes how to configure your router to act as an AppleTalk Remote Access (ARA) server. It includes the following main sections:

This chapter does not describe how to configure or use the client Macintosh. Refer to the Apple Computer, Inc. *Apple Remote Access Client User's Guide* and the *Apple Remote Access Personal Server User's Guide* for information about how to set up and use the ARA software on your Macintosh.

For a complete description of the commands in this chapter, refer to the *Cisco IOS Terminal Services Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release.

ARA Overview

The Cisco implementation of ARA gives Macintosh users direct access to information and resources in remote AppleTalk networks over standard telephone lines. For example, if you have a PowerBook at home and need to get a file from your Macintosh at the office, ARA software can make the connection between your home and office computers over telephone lines.

You can configure your router to act as an ARA server by enabling AppleTalk and ARA protocol on physical terminal (TTY) or virtual terminal lines. Configuring your router to act as an ARA server allows remote Macintosh users to dial in, become a network node, and connect to devices on other networks.

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

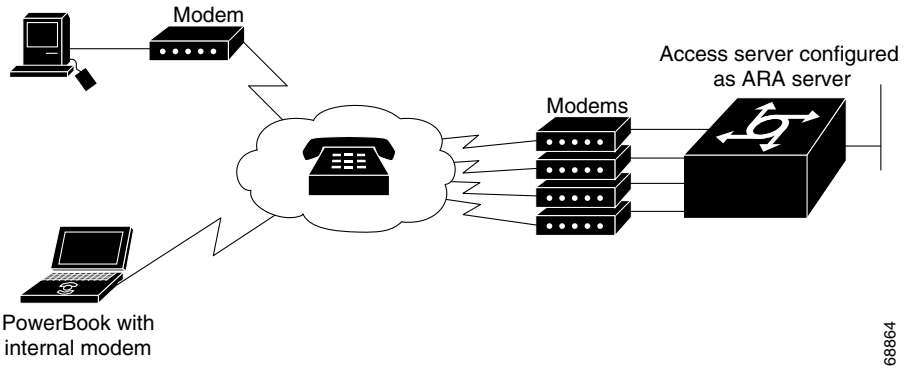
ARA protocol support is transparent to the Macintosh end user. Macintosh users can also use Serial Line Internet Protocol (SLIP) to access remote IP network resources and PPP to access both AppleTalk and IP resources.

The following Macintosh and Cisco IOS software support is required for ARA connectivity:

- Macintosh running ARA software and a connection control language (CCL) script.
- Router configured as an ARA server.

Figure 1 shows how your router can act as an ARA server between remote Macintosh computers (in Figure 1, a Power Macintosh and a PowerBook) and devices on another network.

Figure 1 ARA Configuration Overview



ARA Configuration Task List

To set up the Cisco IOS software to act as an ARA server, perform the tasks described in the following sections:

- Connecting Cables (Required)
- Configuring the Line and the Modem (Required)
- Configuring ARA (Required)
- Configuring ARA to Start Up Automatically (Optional)
- Configuring ARA Security (Optional)
- Connecting to an AppleTalk Network from a Client Running a Different Virtual Terminal Protocol (Optional)

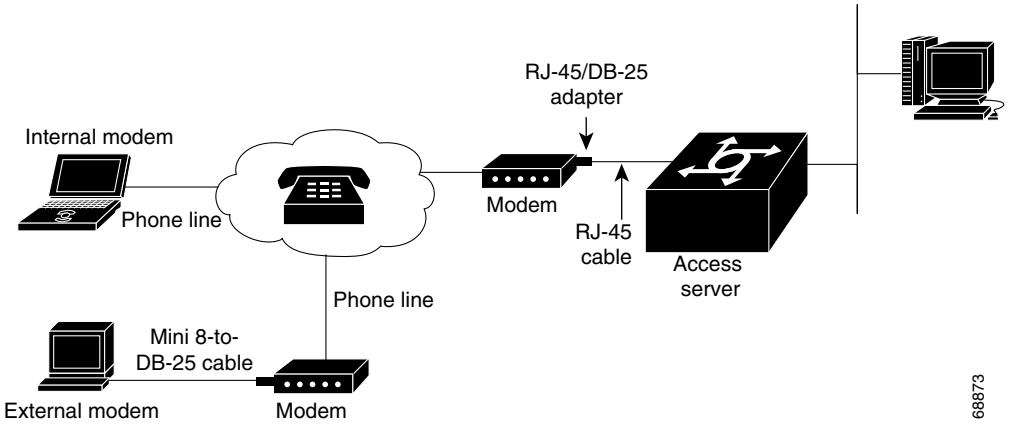
To enable remote clients running PPP to dial in and access AppleTalk resources on a network, you must configure AppleTalk Control Protocol (ATCP). To configure ATCP, refer to "*Configuring Asynchronous SLIP and PPP*" in the *Cisco IOS Dial Technologies Configuration Guide*.

The section "Making ARA Connections" later in this chapter provides connection information. Refer to the "Monitoring an ARA Server," "Monitoring the AppleTalk Network," and "Troubleshooting ARA Connections" sections for information about maintaining and troubleshooting the ARA server and AppleTalk network. The section "ARA Configuration and Connection Examples" provides configuration examples.

Connecting Cables

Figure 2 shows how to connect a Macintosh using internal and external modems.

Figure 2 ARA Server Cabling and Connections



Use the MMOD version of the RJ-45-to-DB-25 adapter (labeled "Modem" if the adapter is from Cisco) to connect a "rolled" RJ-45 cable from the router to the modem. Use a high-speed modem cable with hardware flow control to connect a modem to your Macintosh (see the user documentation for your modem for more specific information).

Some Cisco access servers such as the Cisco AS5800 and Cisco AS5300 have internal modems. Therefore there are no modem cables for you to connect.

For more information about connecting cables, see the installation and configuration or product user guide that came with your router.

Configuring the Line and the Modem

To configure the line, perform the following steps:

Step 1 Specify the maximum common line speed for the modem and the access server. The access server supports 4-fold compression of data, so you can use the speeds shown in the following list:

- 115,200 bits per second (bps) for use with modems that support a transmission rate of 28,800
- 57,600 bps for use with modems that support a transmission rate of 14,400
- 38,400 bps for use with modems that support a transmission rate of 9,600

Note See your modem guide to ensure that the modem can support these maximum line speeds.

Step 2 Set hardware flow control. Use the **flowcontrol hardware** command to enable hardware flow control.

Note The Cisco IOS software does not support modems that do not support hardware flow control.

Step 3 Specify your modem control parameters. Use the **modem inout** command to configure the line for both incoming and outgoing calls, or use the **modem dialin** command to configure the line for incoming calls only.

Step 4 Configure security on your dial-in lines. Use the **aaa new-model** command to enable the authentication, authorization, and accounting (AAA) process on the router, the **aaa authentication arap** command to create an authentication list, and the **arap authentication** command to apply the authentication list to a line or set of lines configured for ARA.

For more information about configuring lines and modem control, refer to *Cisco IOS Dial Technologies Configuration Guide*. For information about configuring security, refer to *Cisco IOS Security Configuration Guide*.

Note The **autobaud** command is not supported with ARA and should never be used.

Configuring ARA

To allow ARA connections to pass through the access server or router, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# appletalk routing`	Enables AppleTalk.[1]
Step 2	`Router(config)# arap network` [*network-number*] [*zone-name*]	Creates a new network or zone for ARA clients when they dial in. The *network-number* argument must be a unique network number.
Step 3	`Router(config-if)# appletalk send-rtmps`	In interface configuration mode, ensures that a new internal network is advertised by enabling the Routing Table Maintenance Protocol (RTMP). You need to configure an AppleTalk interface using the discovery mode in the Cisco IOS software. To do so, an interface on the router must be connected to a network that has at least one other router configured for AppleTalk.
Step 4	`Router(config-if)# appletalk routing`	Returns to global configuration mode and turns on AppleTalk routing.
Step 5	`Router(config)# line` [**tty** | **aux** | **vty**] *line-number* [*ending-line-number*]	Enters line configuration mode.
Step 6	`Router(config-line)# arap enable`	Enables ARA on a line.

1. For more information about configuring AppleTalk, refer to *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

If you discover that an AppleTalk network already exists, the zone and cable range must match the existing configuration. To identify existing cable ranges and zone names, configure the Cisco IOS software for discovery mode. You must manually configure an AppleTalk interface on a segment for which there are no AppleTalk routers. For more information, refer to *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

Configuring ARA to Start Up Automatically

Refer to this section after you have configured AppleTalk routing, created an internal ARA network or zone, and enabled ARA. At this point, you can enable optional tasks.

To configure the Cisco IOS software to allow an ARA session to start automatically, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# autoselect {arap | ppp | slip | during-login}`	Configures a line to automatically start an ARA session.
Step 2	`Router(config)# line x`	Enters line configuration mode (x = the line you want to configure in Step 3).
Step 3	`Router(config-line)# arap dedicated`	Enters line configuration mode and dedicate a line to function only as an ARA connection.
Step 4	`Router(config-line)# arap timelimit [minutes]`	Sets the maximum length of an ARA session for a line. The default is unlimited length connections.
Step 5	`Router(config-line)# arap warningtime [minutes]`	Determines when a disconnect warning message is displayed, in number of minutes before the line is disconnected. This command is valid only when a session time limit is set.

The **autoselect** command permits the router to start an ARA session automatically when it detects the start character for an Appletalk Remote Accesses Protocol (ARAP) packet. The Cisco IOS software detects either a Return character, which is the start character for an EXEC session, or the start character for the ARA protocol. By entering the **autoselect** command with the **during-login** keyword, you can display the username or password prompt without pressing the Return key. While the username or password prompts are displayed, you can choose to answer these prompts or to start sending packets from an autoselected protocol.

Normally a router avoids line and modem noise by clearing the initial data received within the first few seconds. However, when the autoselect PPP feature is configured, the router flushes characters initially received and then waits for more traffic. This flush causes timeout problems with applications that send only one carriage return. To ensure that the input data sent by a modem or other asynchronous device is not lost after line activation, enter the **flush-at-activation** line configuration command.

For information about using ARA with TACACS, Extended TACACS, and AAA/TACACS+, refer to *Cisco IOS Security Configuration Guide*.

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

To customize the AppleTalk configuration even further, you can perform the following additional tasks:

- Disable checksum generation and verification.
- Configure MacIP.

For more information about these and other tasks you can perform to customize your AppleTalk configuration, refer to *Cisco IOS AppleTalk and Novell IPX Configuration Guide.*

Configuring ARA Security

The following three types of security can be used with ARA:

- ARA Server Security, including required manual password entry, limited network visibility, and no guest access.
- Local or Remote Security Database, including username and password authentication and access lists.
- TACACS and TACACS+ Security for ARA, including TACACS, AAA/TACACS+, and Kerberos.

The following sections describe these tasks. Refer to *Cisco IOS Security Command Reference* for information about commands listed in these tasks.

ARA Server Security

Security features that are specific to the ARA protocol are described in the following sections:

- Requiring Manual Password Entry
- Limiting Network Visibility
- Disallowing Guests

Requiring Manual Password Entry

You can control access by requiring users to enter their password manually at the time they log in. To force manual password entry, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# arap require-manual-password`	Requires manual password entry.

Limiting Network Visibility

You can control Macintosh access to zones and networks by using **arap** commands to reference access control lists configured using AppleTalk **access-list** commands.

To control which zones the Macintosh user can see, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# arap zonelist` *zone-access-list-number*	Limits the zones the Macintosh user sees.

To control traffic from the Macintosh to networks, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# arap net-access-list` *net-access-list-number*	Controls access to networks.

Disallowing Guests

A guest is a user that connects to the network without the need to give a name or a password. To prohibit Macintosh guests from logging in through the router, use the following command in line configuration mode. Use the optional **if-needed** argument to allow users to log in as guests if they are already authenticated with a username or password.

Command	Purpose
`Router(config-line)# arap noguest [if-needed]`	Prohibits guests from logging in to the ARA network.

Note Do not use the **arap noguest** command if you are using modified CCL scripts and the **login tacacs** command.

Local or Remote Security Database

To prevent unauthenticated users from accessing your network resources, you configure a username and password database. This database can be local on the router or can be stored on a remote security server (a PC or UNIX computer set up with a security database). To configure the Cisco IOS software to support either local or remote authentication, perform the tasks described in the following sections:

- Configuring Local Username Authentication (As Required)
- Enabling Remote TACACS or TACACS+ Server Authentication (As Required)

Configuring Local Username Authentication

To configure internal username authentication, use the following command in global configuration mode. Enter this information for each supported user.

Command	Purpose
`Router(config)# username` *name* `[user-maxlinks` *link-number*`] password` *secret*	Specifies a username and password. Optionally, you can specify the maximum number of connections a user can establish. To use the **user-maxlinks** keyword, you must also use the **aaa authorization network default local** command, and PPP encapsulation and name authentication on all the interfaces the user will be accessing.

When users try to log in to the access server, username and password prompts require them to authenticate themselves before they can have access to the router or the network.

Enabling Remote TACACS or TACACS+ Server Authentication

To enable the Cisco IOS software to use a remote TACACS or TACACS+ authentication database, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# tacacs-server host {hostname | ip-address}`	Specifies the IP address or the host name of the remote TACACS+ server host. This host is typically a UNIX system running TACACS+ software.
Step 2	`Router(config)# tacacs-server key shared-secret-text-string`	Specifies a shared secret text string used between the router and the TACACS+ server. The router and TACACS+ server use this text string to encrypt passwords and exchange responses.

After you specify these commands in the Cisco IOS software, you must populate the remote username database to all users to whom you want to provide network access. When users try to log in to the router, username and password prompts require them to authenticate themselves before they can have access to the router or the network.

TACACS and TACACS+ Security for ARA

You can prevent unauthenticated users from accessing your network resources using the following security mechanisms:

- TACACS and AAA/TACACS+ user authentication, with username and password information stored on a TACACS or TACACS+ server
- Kerberos, which is configured through the AAA facility

For more information about each of these security mechanisms, refer to *Cisco IOS Security Configuration Guide*.

To configure TACACS and TACACS+ security to authenticate clients that are using ARA to dial in, perform the tasks described in the following sections:

- Enabling Standard and Extended TACACS for ARA Authentication (Required)
- Enabling AAA/TACACS+ for ARA Authentication (Required)
- Modifying Scripts to Support a Standard EXEC Security Dialog (Optional)—This modification is only necessary if you are running standard TACACS on both your router and your TACACS server.

Enabling Standard and Extended TACACS for ARA Authentication

To use extended TACACS, you must already have set up an extended TACACS server using the Cisco extended TACACS server software, available from the ftp.cisco.com directory. Refer to the README file in this directory for more information. The following two authentication methods are used with standard TACACS:

- You issue the **arap use-tacacs** command. The remote user logs in by entering the appropriate username at the ARA username prompt and password at the password prompt.
- You issue the **arap use-tacacs** command and the **single-line** keyword. The remote user logs in by entering *username*password* at the ARA username prompt, and **arap** at the password prompt.

Note The **arap use-tacacs** command provides TACACS security without the need to modify CCL scripts and respond to dialog boxes. The use of scripts is still a supported feature, and is described in the section "Modifying Scripts to Support a Standard EXEC Security Dialog" later in this chapter.

To configure the router to authenticate using TACACS, use the following commands in line configuration mode:

	Command	Purpose
Step 1	`Router(config-line)# arap use-tacacs [single-line]`	Enables TACACS under ARA.
Step 2	`Router(config-line)# login tacacs`	Enables login authentication using TACACS.

For an example of enabling TACACS for ARA authentication, refer to the section "ARA Configuration and Connection Examples" later in this chapter.

Enabling AAA/TACACS+ for ARA Authentication

To enable TACACS+ authentication for ARA sessions, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# aaa new-model`	Enables the AAA function in the Cisco IOS software.
Step 2	`Router(config)# aaa authentication arap | login {default | list-name} method1 [...[method4]]`	Creates an authentication list that you later apply to lines configured for ARA sessions or when you log in to the router.
Step 3	`Router(config)# line [tty] line-number [ending-line-number]`	Enters line configuration mode.
Step 4	`Router(config-line)# arap authentication {default | list-name}`	Applies an ARA authentication list to lines configured for ARA.
Step 5	`Router(config-line)# login authentication {default | list-name}`	Applies a login authentication list to lines that users can log in to.

Modifying Scripts to Support a Standard EXEC Security Dialog

This section describes how to modify your CCL script to work with TACACS security and how to configure a line to use a TACACS server for user authentication.

Caution Because of the underlying structure of the ARA protocol, modem-layer error control is disabled during the exchange of username and password. This condition makes the exchange highly susceptible to line noise, especially at higher baud rates enabled by V.34 modems. For this reason, we do not recommend the use of modified scripts and encourage users to either upgrade to later versions of TACACS or to use the **arap use-tacacs single-line** command.

For information on how to use TACACS without modifying scripts, refer to the section "Enabling Standard and Extended TACACS for ARA Authentication" earlier in this chapter. For information about the **arap** commands, refer to *Cisco IOS Terminal Services Command Reference*.

If you are currently using modified CCL scripts and want to migrate to nonmodified scripts, refer to the section "Modified and Unmodified CCL Scripts Sample Commands" later in this chapter for information on how to use both in the same environment.

For several popular modems, Cisco provides CCL files that you can use as examples to modify your CCL scripts to support TACACS security. This section explains how to use the CCL files provided by Cisco with TACACS security.

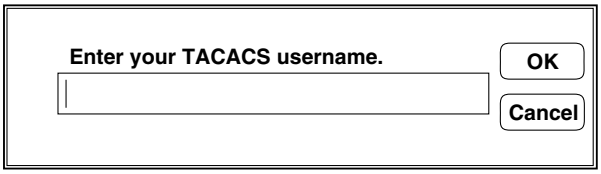
For more information about creating modem initialization scripts, use the ARA Modem Toolkit provided through the AppleTalk Programmers and Developers Association (APDA); it provides both syntax checking and a script tester.

The Macintosh client uses ARA CCL scripts to establish point-to-point links with the modem to the AppleTalk network. When the connection has been established, the script ends and ARA is activated. TACACS authentication occurs after the connection is established and the ARA script ends, but before the ARAP protocol becomes active.

Insert TACACS logic just before the end of a script. The CCL TACACS logic performs the following user authentication tasks:

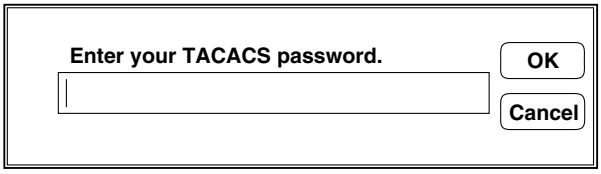
1. When the "Username:" prompt is received from the router, the TACACS server queries the user for a username, as shown in Figure 3.

Figure 3 TACACS Login Screen on the Macintosh Computer



2. When the "Password:" prompt is received from the router, the TACACS server queries the user for a password, as shown in Figure 4.

Figure 4 TACACS Password Screen on the Macintosh Computer



3. After a successful login, indicated by an EXEC prompt, the **arap** EXEC command is executed.
4. The script ends and ARA is activated on the client.

   CCL scripts control logical flow by jumping to labels. The labels are the numbers 1 through 128 and are not necessarily in sequential order in script files. The TACACS logic in the Cisco IOS software CCL files has label numbers from 100 through 127. In most environments, you can copy the complete TACACS logic from a sample file.

To create a new TACACS CCL file, perform the following steps:

Step 1 Copy the TACACS logic from a sample CCL script into the new CCL script.

In most cases, you can insert the TACACS logic at the appropriate place in your CCL script. The one case that requires extra attention is when the original CCL script has labels that conflict with the logic in the new file. The labels must be resolved on a case-by-case basis, usually by changing the label numbers used in the original CCL script. Be sure to read the manual that comes with the ARA Modem Toolkit before beginning.

Step 2 Locate the logical end of the CCL script and insert the **jump 100** command.

You can locate the logical end of the script by following its flow. Most scripts have the following basic structure:

- Initialize the modem.
- Dial the number.
- Exit.

The characteristic logical end of the script is as follows:

```
@label N
! N is any integer between 1 and 128.
if ANSWER N+1
! If we're answering the phone, jump directly
! to the label N+1.
pause 30
! We're not answering the phone, therefore we
! must be calling. Wait three seconds for the
! modems to sync up.
@label N+1
exit 0
! Quit and start up ARA.
```

It is common in this case to replace “pause 30” with “jump 100.” In fact, this replacement is usually the only change made to the logic of the original CCL script.

Refer to *Cisco IOS Dial Technologies Configuration Guide* for information about configuring a line to support your modem.

Enabling Kerberos Security for ARA Authentication

You can use Kerberos as an authentication method within ARA sessions. To do so, you configure Kerberos using the AAA/TACACS+ facility in the Cisco IOS software.

To enable Kerberos security, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# kerberos local-realm {kerberos-realm}`	Defines the name of the Kerberos realm in which the router is located.
Step 2	`Router(config)# kerberos realm {dns-domain | dns-host} kerberos-realm`	Defines the DNS domain of the Kerberos realm in which the router is located.
Step 3	`Router> show kerberos creds`	Displays the contents of your credentials cache.
Step 4	`Router> clear kerberos creds`	Deletes the contents of your credentials cache.

For more information about Kerberos authentication, refer to the *Cisco IOS Security Configuration Guide*.

Using Access Lists to Control Access to AppleTalk Networks

An access list is a list of AppleTalk network numbers or zones that is maintained by the Cisco IOS software and used to control access to or from specific zones or networks. For more information about AppleTalk access lists, refer to *Cisco IOS AppleTalk and Novell IPX Configuration Guide*.

Connecting to an AppleTalk Network from a Client Running a Different Virtual Terminal Protocol

ARA can run on any point-to-point link, such as a Public Switched Telephone Network (PSTN) or an X.25 WAN. This capability permits remote Macintosh users to dial in to a remote network and access AppleTalk services (such as file sharing and printing). For example, you can enable a Macintosh client on the remote side of an X.25 WAN to connect to an AppleTalk network through the router. To do so, you configure a vty on the router so that the client sees one of two scenarios:

- A client clicks **Connect** in an ARA application dialog box and connects to a vty on the router. ARA automatically starts up on the outgoing vty, and the client is connected to the AppleTalk network. This section describes how to configure the Cisco IOS software for this process.
- A client clicks **Connect** in an ARA application dialog box and connects directly through the router to the AppleTalk network. This process is described in the section "Configuring Tunneling of SLIP, PPP, or ARA" in the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" in this publication.

To enable ARA on virtual terminal lines and enable clients running different virtual terminal protocols to connect to an AppleTalk network through the router, use the following commands beginning in global configuration mode. The first four steps are required. The next eight steps are optional.

	Command	Purpose
Step 1	`Router(config)# appletalk routing`	Turns on AppleTalk routing.
Step 2	`Router(config)# arap network` [*network-number*] [*zone-name*]	Creates an internal AppleTalk network.
Step 3	`Router(config)# line vty` *line-number* [*ending-line-number*]	Enters line configuration mode.
Step 4	`Router(config-line)# arap enable`	Enables ARA on a line.
Step 5	`Router(config-line)# autocommand arap`	Configures automatic protocol startup.
Step 6	`Router(config-line)# arap dedicated`	Sets a dedicated ARA line.
Step 7	`Router(config-line)# arap timelimit` [*minutes*]	Sets the session time limit.
Step 8	`Router(config-line)# arap warningtime` [*minutes*]	Sets the disconnect warning time.
Step 9	`Router(config-line)# arap noguest`	Disallows guests.
Step 10	`Router(config-line)# arap require-manual-password`	Requires manual password entry.
Step 11	`Router(config-line)# arap zonelist` *zone-access-list-number*	Limits the zones the Macintosh user sees.
Step 12	`Router(config-line)# arap net-access-list` *net-access-list number*	Controls access to networks.

Making ARA Connections

If you are a Macintosh user, you can use ARA to connect to an AppleTalk network through a Cisco access server. The Cisco IOS Release 10.2 and later release software support ARA 2.0 and ARA 1.0 so that you can remotely dial in through asynchronous network devices using ARA to access AppleTalk services (such as file sharing and printing) elsewhere on the network. For example, you can dial in from an X.25 network and connect to an AppleTalk network through a router. To enable ARA and dial-in access, configure a vty on the router. You can also configure ARA on TTY lines.

Because there are no user commands for connecting to the network from your Macintosh client, the process is not described in this publication. To start a connection in most ARA client packages, you click the **Connect** button from within the client software.

Monitoring an ARA Server

To display information about a running ARA connection, use the following command in privileged EXEC mode (reached by entering the **enable** command and a password at the EXEC prompt):

Command	Purpose
Router# **show arap** [*line-number*]	Displays information about a running ARA connection.

The **show arap** command with no arguments displays a summary of ARA traffic since the router was last booted. The **show arap** command with a specified line number displays information about the connection on that line.

Monitoring the AppleTalk Network

The Cisco IOS software provides several commands that you can use to monitor an AppleTalk network. In addition, you can use Inter-Poll from Apple Computer, which is a tool to verify that a device is configured and operating properly. Use the commands described in this section to monitor an AppleTalk network using both Cisco IOS software commands and Inter-Poll.

To monitor the AppleTalk network, use any of the the following commands in EXEC mode:

Command	Purpose
Router> **show appletalk arp**	Lists the entries in the AppleTalk ARP table.
Router> **show appletalk interface** [**brief**] [*type number*]	Displays AppleTalk-related interface settings.
Router> **show appletalk macip-clients**	Displays the status of all known MacIP clients.
Router> **show appletalk macip-servers**	Displays the status of MacIP servers.
Router> **show appletalk macip-traffic**	Displays statistics about MacIP traffic.
Router> **show appletalk traffic**	Displays the statistics about AppleTalk protocol traffic, including MacIP traffic.
Router> **show appletalk zone** [*zone-name*]	Displays the contents of the zone information table.

Troubleshooting ARA Connections

Use ARA debugging enhancements to troubleshoot one or more asynchronous lines on an access server. These enhancements are supported on all Macintosh terminals and all Cisco routers and access servers that support the AppleTalk software feature set.

Allowing users to specify a single line via an additional parameter for troubleshooting produces the following benefits:

- Focused results—Users get only the information they need.
- Reduced server load—Heavily loaded servers are subject to developing ARAP problems which need to be fixed by debugging. However, debugging itself increases the server work load. By focusing on specific lines, the impact of debugging activity on the server is minimized.
- Targeting flexibility—By being able to debug on just the lines in a group of lines, users can solve problems in rotary groups in which there is no way to specify which line or group of lines a remote user will be assigned.

To enable ARAP debugging, use the following commands beginning in EXEC mode:

	Command	Purpose
Step 1	`Router# debug arap {internal | memory | mnp4 | v42bis}`	Enters debug mode and specifies the type of the debug. To debug internal ARA packets, specify the **internal** keyword. To debug the memory allocated to ARA, specify the **memory** keyword. To debug the serial protocol, specify the **mnp4** keyword. To debug compression, specify the **v42bis** keyword.
Step 2	`Router# debug arap internal [linenum [aux | console | tty | vty]]`	Replaces the *linenum* variable with a single line number. Specifies the target for the debug. Specify the **aux** keyword to debug an auxiliary line, the **console** keyword to debug a primary terminal line, the **tty** keyword to debug a physical terminal asynchronous line, or the **vty** keyword to debug a vty.

To verify if the debug level and target are set correctly, enter the **show debug** command:

```
Router# show debug

AppleTalk Remote Access:
ARAP MNP4 debugging is on for line 7
```

ARAP Debugging Examples

The following example sets ARAP debugging in memory mode on line 7. The **show debug** command confirms the configuration.

```
Router# debug arap mn 7
 ARAP MNP4 debugging is on for line 7
Router# debug arap mn 8
 ARAP MNP4 debugging is on for line 8
Router# debug arap mn 9
 ARAP MNP4 debugging is on for line 9
Router# show debug
```

```
 AppleTalk Remote Access:
   ARAP MNP4 debugging is on for line 7
   ARAP MNP4 debugging is on for line 8
   ARAP MNP4 debugging is on for line 9
```

Note You can debug several lines (for example, lines in a rotary), but you must turn on debugging one line at a time.

The following example sets ARAP debugging in internal mode on line 6, memory mode on line 10, and V.42*bis* compression mode on line 6. The **show debug** command confirms the configuration.

```
Router# debug arap in 6
 ARAP internal packet debugging is on for line 6
Router# debug arap me 10
 ARAP memory debugging is on for line 10
Router# debug arap v 6
 ARAP V.42bis debugging is on for line 6
Router# show debug
 AppleTalk Remote Access:
   ARAP V.42bis debugging is on for line 6
   ARAP internal packet debugging is on for line 6
   ARAP memory debugging is on for line 10
```

The following example sets ARAP debugging for each mode in succession and for all lines. The **show debug** command confirms the configuration.

```
Router# debug arap mnp4
 ARAP MNP4 debugging is on
Router# debug arap internal
 ARAP internal packet debugging is on
Router# debug arap v42bis
 ARAP V.42bis debugging is on
Router# debug arap memory
 ARAP memory debugging is on
Router# show debug
 AppleTalk Remote Access:
   ARAP MNP4 debugging is on
   ARAP V.42bis debugging is on
   ARAP internal packet debugging is on
   ARAP memory debugging is on
Router#
```

The following example sets all debugging (including ARAP debugging) for all modes and for all lines. The **show debug** command confirms the configuration. Note that turning on all debugging utilities can slow down performance.

```
Router# debug all
 This may severely impact network performance. Continue? [confirm] y
 All possible debugging has been turned on
Router# show debug
 "debug all" is in effect.
```

The following example turns off ARAP debugging. The **show debug** command confirms the configuration.

```
Router# undebug all
 All possible debugging has been turned off
Router# show debug
Router#
```

The following example shows debug output for two lines, 2 and 4. The boldfaced portion of this example shows that for line 2, `LA` is the MNP4 acknowledge frame, `31` is the sequence number of the last frame, and `08` is the window size.

```
ARAP MEM TTY 4: arap_getbuffer 94745C
 ARAP MEM TTY 4: arap_datagram_done 7BD324
 MNP4 TTY 4:mnp4_input()
 MNP4 TTY 2:mnp4_input()
 ARAP MEM TTY 2: arap_getbuffer 7BD158
 MNP4 TTY 2:Rcv LA Nr[31] Nk[08]
 ARAP MEM TTY 2: arap_datagram_done 7BD6BC
 MNP4 TTY 4:mnp4_input()
 ARAP SMARTBUF TTY 2: ring end 936C62, start 934ED4, need 58 bytes
 ARAP SMARTBUF TTY 2: new seq 161

 ARAP TTY 4: Received TICKLE
 ARAP TTY 4: --------- ACKing 125 ---------
 ARAP SMARTBUF TTY 2: ring end 936C28, start 934ED4, need 58 bytes
 ARAP SMARTBUF TTY 2: new seq 160
 ARAP SMARTBUF TTY 2: ring end 9342B4, start 9322EC, need 64 bytes
 ARAP SMARTBUF TTY 2: new seq 144
 ARAP SMARTBUF TTY 2: search...
 ARAP SMARTBUF TTY 2: search...
0 ddp; trailing; 1 ddp; trailing; 2 ddp; trailing; 3 ddp; trailing; 4 ddp; trailing; 5
ddp; 6 offset; 7 ddp; trailing; 8 ddp; 9 offset; 10 ddp; trailing; 11 ddp; trailing; 12
ddp; trailing; 13 ddp; trailing; 14 ddp; 15 ddp; trailing; 16 ddpARAP SMAR
@TBUF TTY 2: ring end 936C62, start 934ED4, need 58 bytes
 ARAP SMARTBUF TTY 2: new seq 161

 ARAP TTY 4: Received TICKLE
 ARAP TTY 4: --------- ACKing 125 ---------
 ARAP TTY 2: Received TICKLE
 ARAP TTY 2: --------- ACKing 114 ---------

 V42bis TTY 4: OUT uncomp (12): 0 10 16 33 0 9 1 195 255 255 255 255
 V42bis TTY 4: OUT comp (6): 10 38 229 203 3 0
 V42bis TTY 4: IN comp (6): 205 145 196 79 2 0
 V42bis TTY 4: IN uncomp (12): 0 10 16 143 0 9 0 0 255 255 255 255
 V42bis TTY 4: OUT uncomp (6): 0 4 16 143 0 0
 V42bis TTY 4: OUT comp (6): 182 244 235 0 2 0
 V42bis TTY 4: IN comp (6): 217 111 250 0 2 0
 V42bis TTY 4: IN uncomp (6): 0 4 16 33 0 0
 V42bis TTY 2: IN comp (5): 247 225 15 102 0
 V42bis TTY 2: IN uncomp (12): 0 10 16 132 0 9 255 219 255 255 255 255
 V42bis TTY 2: OUT uncomp (6): 0 4 16 132 0 0
 V42bis TTY 2: OUT comp (6): 126 63 196 65 2 0
.
```

ARA Configuration and Connection Examples

This section contains the following examples of and procedures for ARA configuration:

- ARA Server Configuration Procedure
- Dedicated ARA Line with User Authentication Example
- Autostart Multiple ARA Lines with User Authentication Example
- Telebit T-3000 Modem Setup Procedure
- Modified and Unmodified CCL Scripts Sample Commands
- ARA Router Support Example

- Extended AppleTalk Network Example
- Cable Range Expansion Example
- Extended Network in Discovery Mode Example
- TACACS Username Authentication Example
- TACACS Enabled for ARA Authentication Example
- AppleTalk Network Connection over a Foreign Protocol Example

ARA Server Configuration Procedure

The following sample procedure shows how to set up ARA functionality.

Log in to the router, use the **enable** command to enter your password if one is set, use the **configure** command to enter configuration mode, and add the following commands to your configuration:

```
appletalk routing
arap network 104 ARAP Dialin Zone
interface ethernet 0
 appletalk cable-range 0-0 0.0
! Puts router in discovery mode.
line 5 6
 modem inout
 speed 38400
 arap enabled
 autoselect
```

If you already know the cable range and the zone names you need, include the information in the configuration file. If you do not know this information, perform the following steps to use the discovery mode to allow the Cisco IOS software to learn about the AppleTalk network:

Step 1 Permit the Cisco IOS software to monitor the line for a few minutes.

Step 2 Log in and enter configuration mode.

Step 3 Display the configuration again (using the **more nvram:startup-config** command).

Step 4 Note the **appletalk cable-range** and **appletalk zone** variables.

Step 5 Manually add the information in those two entries and add any user accounts:

```
appletalk cable-range 105-105 105.222
appletalk zone Marketing Lab
username arauser password arapasswd
! Add as many users as you need.
```

Step 6 Save the configuration.

Step 7 Display the configuration again (using the **more nvram:startup-config** command) to make sure the configuration is correct.

Dedicated ARA Line with User Authentication Example

The following example configures line 2 as a dedicated ARA line with user authentication information on the ARA server; guests are not allowed to make ARA sessions:

```
username jsmith password woof
line 2
 arap dedicated
 arap noguest
```

Autostart Multiple ARA Lines with User Authentication Example

The following example enables ARA on lines 2 through 16. Username authentication is configured on the ARA server, and the lines are configured to automatically start an ARA session when an ARA user on a Macintosh attempts a connection.

```
username jsmith password woof
line 2 16
 autoselect
 arap enabled
 arap noguest
```

Telebit T-3000 Modem Setup Procedure

To set up a Telebit T-3000 modem that attaches to a router, which supports hardware flow control, perform the following steps. The Macintosh will use a CCL script to configure the attached modem.

Caution When you configure modems for ARA, turn off MNP4 error correction because it can cause connection failures for ARA 1.0 clients. For dedicated ARA lines, it is sufficient to turn off error correction completely in the modem; for multiuse lines it is preferable to leave all forms of non-MNP4 error correction enabled so that users of other protocols can achieve error-corrected connections. This restriction does not apply to installations that only receive calls from ARAP 2.0 clients.

Step 1 Start with the modem at factory defaults. (The preferred configuration for hardware flow control is AT&F9.) Use the **direct** command if you have a terminal attached to the modem, or use the T/D Reset sequence described in the Telebit T-3000 manual to reset the modem to the &F9 defaults.

Step 2 Attach a hardware flow control-capable cable between the modem and the device with which you are configuring the modem. (At this point, the modem is in hardware flow control mode, with autobaud-rate-recognition, and can detect your speed from 300 to 38,400 bps at 8-N-1. However, the modem must receive the flow control signals from the device to which you have the modem attached.)

Step 3 Send the modem the following AT commands:

ATS51=6 E0 Q1 S0=2 &D3 &R3 S58=2 &W

This sequence directs the modem to perform the following tasks:

- Lock your DTE interface speed to 38,400 bps.
- Turn “command echo” off.
- Do not send any result codes.
- Auto-answer on the second ring (Germany requires this setting, but elsewhere you can set it to answer on the first ring with “s0=1”).
- When data terminal ready (DTR) is toggled, reset to the settings in NVRAM.
- Clear To Send (CTS) is always enabled if hardware flow control is disabled.
- Use full-duplex request to send/clear to send (RTS/CTS) flow control.
- Write these settings to NVRAM.

Step 4 At this point, if you press the Return key or enter characters, no characters appear on your screen because the result codes are turned off. You can determine whether the modem is working by getting a list of its configuration registers using the AT command **AT&V.**

Step 5 After the modem is configured, connect it to the router with a modem-to-RJ-45 adapter and an RJ-45 cable to the lines that you plan to use.

The following Cisco IOS commands are compatible with the Telebit 3000 settings described in this section:

```
line 1 8
 arap enable
 autoselect
 no escape-character
 flowcontrol hardware
 modem dialin
 speed 38400
```

Modified and Unmodified CCL Scripts Sample Commands

If you are using modified CCL scripts and want to migrate to nonmodified scripts, you can set your system to accept logins using both modified CCL and unmodified scripts. Use the following commands in line configuration mode:

```
 autoselect arap
 autoselect during-login
 arap noguest if-needed
```

ARA Router Support Example

The following example configures the router for ARA support, as described in the comments (lines beginning with an exclamation point [!]):

```
! Enable AppleTalk on the router.
appletalk routing
!
interface Ethernet 0
 ip address 172.30.1.1 255.255.255.0
!
! On interface Ethernet 0, assign network number 103 to the physical cable and
! assign zone name "Marketing Lab" to the interface. Assign a zone name if
! you are creating a new AppleTalk internet. If the internet already exists,
! the zone and cable range must match exactly, or you can leave the cable
! range at 0 to enter discovery mode. The suggested AppleTalk address for the interface in
! this example is 103.1.
interface Ethernet 0
 appletalk cable-range 103-103 103.1
 appletalk zone Marketing Lab
! Configure a username and password for the router.
username jake password sesame
! On lines 4 through 8, InOut modems are specified, the lines are configured
! to automatically start an EXEC session or enable AppleTalk, AppleTalk Remote
! Access Protocol is enabled, the modem speed is specified as 38400 bps, and
! hardware flow control is enabled.
```

```
line 4 8
 modem InOut
 autoselect
 arap enabled
 speed 38400
 flowcontrol hardware
```

Note You must set your terminal emulator to match the speed that you set for the line.

Extended AppleTalk Network Example

The following example configures the interface for an extended AppleTalk network. It defines the zones named Orange and Brown. The cable range of 1 allows compatibility with nonextended AppleTalk networks.

```
appletalk routing
interface ethernet 0
 appletalk cable-range 1-1
 appletalk zone Orange
 appletalk zone Brown
```

Cable Range Expansion Example

The following example changes the cable range and reenters the zone name. The initial configuration is as follows:

```
 appletalk cable-range 100-103
 appletalk zone Twilight Zone
```

The cable range is expanded as follows:

```
appletalk cable-range 100-109
```

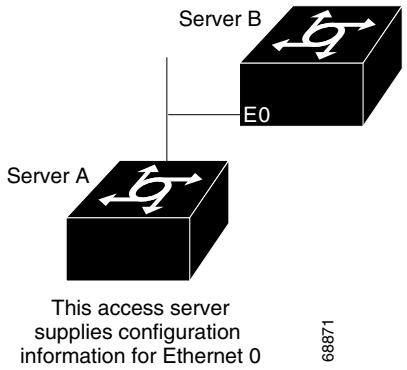
At this point, you must reenter the zone name as follows:

```
appletalk zone Twilight Zone
```

Extended Network in Discovery Mode Example

The following example configures an extended network in discovery mode. In Figure 5, the access server named Server A provides the zone and network number information to the interface when it starts.

Figure 5 Discovery Mode



The following example configures an extended network in discovery mode:

```
appletalk routing
interface ethernet 0
 appletalk cable-range 0-0 0.0
```

TACACS Username Authentication Example

The following example for TACACS and Extended TACACS configures line 1 for ARA and username authentication on a TACACS server:

```
line 1
 login tacacs
 arap enable
```

The following example configures AAA/TACACS+ on line 1 for ARA and username authentication on a TACACS server:

```
line 1
 login authentication
 arap authentication
```

TACACS Enabled for ARA Authentication Example

The following example shows regular TACACS enabled for ARA authentication:

```
line 3
 arap use-tacacs
```

The following example shows AAA/TACACS+ enabled for ARA authentication:

```
line 3
 aaa authentication arap
```

AppleTalk Network Connection over a Foreign Protocol Example

The following example enables a Macintosh client running ARA on a remote network to connect across an X.25 network, through the router, to an AppleTalk network. In this example, virtual terminal lines 0 through 19 are configured for ARA:

```
appletalk routing
```

```
line vty 0 19
 arap enable
 autocommand arap
 arap dedicated
 arap timelimit 45
 arap warningtime 5
 arap noguest
 arap require-manual-password
 arap net-access-list 611
```

The Macintosh client connects to any vty from 0 through 19. When the EXEC prompt appears, ARA begins automatically on the line (because of the **autocommand arap** command). The virtual terminal lines 0 through 19 are dedicated to ARA dial-in clients, and those clients have a 45-minute time limit. Five minutes before the line is disconnected, a warning message appears indicating that the session will be disconnected. Guest access is denied, and manual password entry is required. The AppleTalk access list 611 has been applied to the virtual terminal lines, meaning that access to other networks through these virtual terminal lines has been limited.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

Configuring the Cisco PAD Facility for X.25 Connections

This chapter describes how to use the internal packet assembler/disassembler (PAD) facility to make connections with remote devices over the X.25 protocol. This chapter includes the following sections:

Table 1 in this chapter summarizes the X.3 PAD parameters that you can set. For a complete description of each X.3 parameter supported by the standard X.28 mode or Cisco PAD EXEC user interface, see the appendix "X.3 PAD Parameters" at the end of this publication.

For a complete description of the commands in this chapter, refer to the *Cisco IOS Terminal Services Command Reference*, Release 12.2. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the "Using Cisco IOS Software" chapter.

PAD Connection Overview

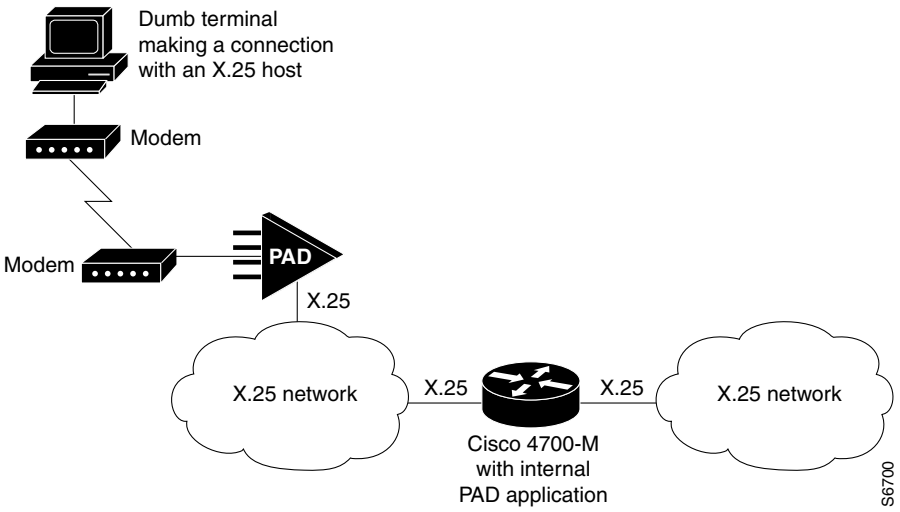
PADs are configured to enable X.25 connections between network devices. A PAD is a device that receives a character stream from one or more terminals, assembles the character stream into packets, and sends the data packets out to a host. A PAD can also do the reverse. It can take data packets from a network host and translate them into a character stream that can be understood by the terminals. A PAD

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

is defined by Recommendations X.3, X.28, and X.29 of the International Telecommunication Union Telecommunication Standardization Sector (ITU-T). (The ITU supersedes the Consultative Committee for International Telegraph and Telephone, or CCITT).

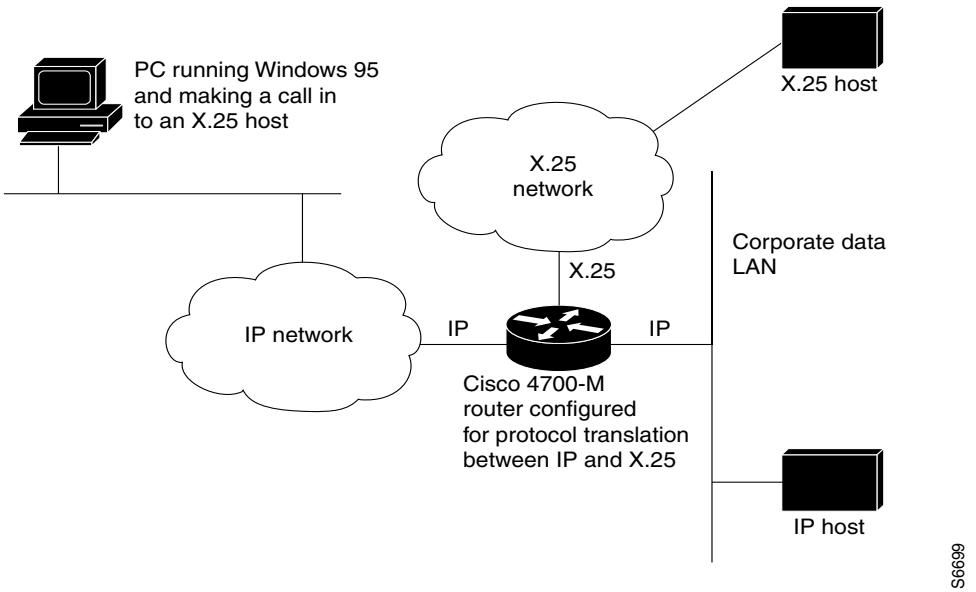
Figure 1 shows a remote X.25 user placing a call through an X.25 switched network to the internal PAD application on a Cisco 4700-M router, and to an X.25 host located inside a corporate data center.

Figure 1 Standard X.25 Connection Between a Dumb Terminal and an X.25 Host



PADs can also be configured to work with a protocol translation application. Figure 2 shows an example of a remote PC placing an analog modem call to an IP network, connecting to a Cisco 4500-M router, and allowing its IP packets to undergo IP-to-X.25 protocol translation. The remote PC, in turn, communicates with an internal PAD device in the Cisco router and establishes a connection with an X.25 host.

Figure 2 PC Dialing In to an X.25 Host Using Protocol Translation



Cisco IOS offers two ways of connecting to a PAD: using the **pad** EXEC user interface command to initiate an outgoing connection to a PAD, and using the **x28** EXEC command to access the Cisco universal X.28 PAD user emulation mode.

In X.28 PAD user emulation mode, you can perform the same functions available from the Cisco **pad** EXEC user interface; however, X.28 PAD user emulation mode adds functionality such as the ability to exchange PAD signals across an X.25 network, and is useful for connecting to systems using software designed to interact with an X.28 PAD. X.28 PAD user emulation mode is also useful when a reverse connection requires packetization according to the X.29 parameters.

Cisco PAD EXEC User Interface Connections

The Cisco IOS **pad** EXEC user interface initiates an outgoing call to a PAD host and in most cases is the preferred PAD connection method. You can have multiple PAD connections open at one time. Options are available for pausing and resuming connections, and setting X.3 PAD parameters at the command line.

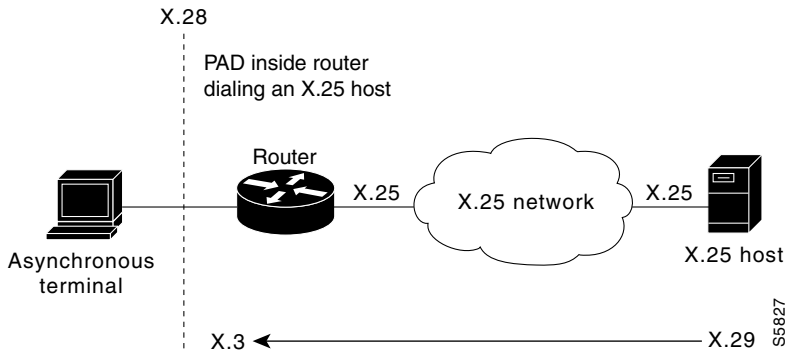
Cisco Universal X.28 PAD Emulation Mode

The Cisco IOS software provides a universal X.28 user emulation mode that enables you to interact with and control the PAD. X.28 emulation effectively turns the Cisco router into an X.28-compliant PAD device that provides a standard user interface between a DTE device and a PAD.

For asynchronous devices such as terminals or modems to access an X.25 network host, the packets from the device must be assembled or disassembled by a PAD. Using standard X.28 commands from the PAD, calls can be made into an X.25 network, X.3 PAD parameters can be set, or calls can be reset.

X.3 is the ITU-T recommendation that defines various PAD parameters used in X.25 networks. X.3 PAD parameters are internal variables that define the operation of a PAD. For example, parameter 9 is the crpad parameter. It determines the number of bytes to add after a carriage return. X.3 parameters can also be set by a remote X.25 host using X.29. (See Figure 3.)

Figure 3 Asynchronous Device Dialing In to an X.25 Host over an X.25 Network



Note Most Cisco routers have internal PAD devices. Use the Feature Navigator on Cisco.com to determine which software supports PAD connections.

X.28 enables PAD system administrators to dial in to X.25 networks or set PAD parameters using the X.28 standard user interface. This standard interface is commonly used in many European countries. It adheres to the X.25 ITU-T standards.

The X.28 interface is designed for asynchronous devices that require X.25 transport to access a remote or native asynchronous or synchronous host application. For example, dialup applications can use the X.28 interface to access a remote X.25 host. X.28 PAD calls are often used by banks to support applications in the "back office" such as ATM machines, point of sales authorization devices, and alarm systems. An ATM machine may have an asynchronous connection to an alarm host and a Cisco router. When the alarm is tripped, the alarm sends a distress call to the authorities via the Cisco router and an X.28 PAD call.

Cisco X.28 PAD calls can be transported over a public packet network, a private X.25 network, the Internet, a private IP-based network, or a Frame Relay network. X.28 PAD can also be used with protocol translation. Protocol translation and virtual asynchronous interfaces enable users to bidirectionally access an X.25 application with the PAD service or other protocols such as Digital, local-area transport (LAT), and TCP.

X.3 PAD EXEC User Interface Configuration Task List

To connect to a PAD using the EXEC user interface, perform the following tasks:

- Making a PAD Connection (Required)
- Switching Between Connections (Optional)
- Exiting a PAD Session (Optional)
- Monitoring X.25 PAD Connections (Optional)
- Setting X.3 PAD Parameters(Optional)

Making a PAD Connection

To log in to a PAD, use the following command in EXEC mode:

Command	Purpose
`Router> pad {x121-address | hostname} [/cud text] [/debug] [/profile name] [/quiet message] [/reverse] [/use-map]`	Logs in to a PAD.

You can exit a connection and return to the user EXEC prompt at any point.

To open a new connection, first exit the current connection by entering the escape sequence (Ctrl-Shift-6 then x [Ctrl^x] by default) to return to the EXEC prompt.

Switching Between Connections

You can have several concurrent sessions open and switch between them. The number of sessions that can be open is defined by the **session-limit** command, which is described in the *Cisco IOS Terminal Services Command Reference*, Release 12.2.

To switch between sessions by escaping one session and resuming a previously opened session, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`Router>` **`Ctrl-Shift-6`** then **`x`** (**`Ctrl^x`**) by default	Escapes the current connection, if you have one open, and returns to EXEC mode.
Step 2	`Router>` **`where`**	From EXEC mode, lists the open sessions. All open sessions associated with the current terminal line are displayed.
Step 3	`Router>` **`resume`** [*`connection`*] [*`keyword`*]	Makes the connection using the session number displayed by the **where** command.

Note The Ctrl^x, **where**, and **resume** commands are available with all supported connection protocols.

Exiting a PAD Session

To exit a PAD session, enter the escape sequence (Ctrl-Shift-6 then x [Ctrl^x] by default) and enter the **disconnect** command at the EXEC prompt. You can also log out of the remote system by entering the command specific to that system (such as **exit**, **logout**, **quit**, **close**, or **disconnect**).

Monitoring X.25 PAD Connections

To display information about current open connections, use the following command in user EXEC mode:

Command	Purpose
`Router>` **`show x25 pad`**	Displays information about X.25 PAD connections that are open.

The information displayed by **show x25 pad** includes packet transmissions, X.3 parameter settings, and the current status of virtual circuits. The information displayed will help you set and change PAD parameters (see the section "X.3 Parameter Customization Example" for an example).

Setting X.3 PAD Parameters

To set X.3 PAD parameters, use one of the following commands in EXEC mode:

Command	Purpose
`Router>` **`resume`** [*`connection`*] [**`/set`** *`parameter:value`*] or `Router>` **`x3`** *`parameter:value`*	Sets X.3 PAD parameters.

Table 1 summarizes the X.3 PAD Parameters supported on Cisco devices. See the "X.3 PAD Parameters" appendix in this publication for more complete information about these parameters. Refer to the "ASCII Character Set and Hex Values" appendix in the *Cisco IOS Configuration Fundamentals Command Reference*, Release 12.2, for a list of ASCII characters.

Table 1 Supported X.3 PAD Parameters

Parameter Number	ITU-T Parameter Name	ITU-T X.3 and Cisco Values
1	PAD recall using a character	Minimum value: 0; maximum value: 126; X.28 PAD user emulation mode default: 1. **Note** Not supported by PAD EXEC user interface.
2	Echo	Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode default: 1.
3	Selection of data forwarding character	Minimum value: 0; maximum value: 255; PAD EXEC mode default: 2 (CR); X.28 PAD user emulation mode default: 126 (~).
4	Selection of idle timer delay	Minimum value: 0; maximum value: 255; PAD EXEC mode default: 1; X.28 PAD user emulation mode default: 0.
5	Ancillary device control	Minimum value: 0; maximum value: 2; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 1.
6	Control of PAD service signals	Minimum value: 0; maximum value: 255; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 2. **Note** Not supported by PAD EXEC user interface.
7	Action upon receipt of a BREAK signal	Minimum value: 0; maximum value: 31; PAD EXEC mode default: 4; X.28 PAD user emulation mode default: 2.
8	Discard output	Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode default: 0.
9	Padding after Return	Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0.
10	Line folding	Not supported.
11	DTE speed (binary speed of start-stop mode DTE)	Minimum value: 0; maximum value: 18; PAD EXEC mode and X.28 PAD user emulation mode default: 14.
12	Flow control of the PAD by the start-stop DTE	Minimum value: 0; maximum value: 1; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 1.
13	Line feed insertion (after a Return)	Minimum value: 0; maximum value: 7; PAD EXEC mode and X.28 PAD user emulation mode default: 0.
14	Line feed padding	Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0.
15	Editing	Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Table 1 Supported X.3 PAD Parameters (continued)

Parameter Number	ITU-T Parameter Name	ITU-T X.3 and Cisco Values
16	Character delete	Minimum value: 0; maximum value: 127; PAD EXEC mode and X.28 PAD user emulation mode default: 127 (DEL).
17	Line delete	Minimum value: 0; maximum value: 127; PAD EXEC mode default: 21 (NAK or Ctrl-U); X.28 PAD user emulation mode default: 24 (CAN or Ctrl-X).
18	Line display	Minimum value: 0; maximum value: 127; PAD EXEC mode and X.28 PAD user emulation mode default: 18 (DC2 or Ctrl-R).
19	Editing PAD service signals	Minimum value: 0; maximum value: 126; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 2. **Note** Not supported by PAD EXEC user interface.
20	Echo mask	Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0. **Note** Not supported by PAD EXEC user interface.
21	Parity treatment	Minimum value: 0; maximum value: 4; PAD EXEC mode and X.28 PAD user emulation mode default: 0. **Note** For additional values that can be selected for parameter 21, see Table 23 in this guide. To select parity treatment to conform to the French Transpac public switched data network and its technical specification and utilization of networks standards (STUR), see Table 24 in this guide.
22	Page wait	Not supported.

X.28 PAD Emulation Configuration Task List

To use the X.28 PAD mode, perform the following tasks as needed:

- Accessing X.28 Mode and Setting Options (Required)
- Exchanging PAD Command Signals (Optional)
- Customizing X.3 Parameters (Optional)
- Accepting Reverse or Bidirectional X.25 Connections (Optional)
- Setting PAD French Language Service Signals (Optional)

The section "Cisco Universal X.28 PAD Emulation Mode Examples" provides examples of making X.28 PAD connections.

Accessing X.28 Mode and Setting Options

To access the Cisco IOS universal X.28 emulation mode, use the **x28** EXEC command. This mode can also be accessed with the **autocommand** line configuration command. The **autocommand** command can be assigned to a particular line, range of lines, or login user ID. In this case, when a user connects to the line, the user sees an X.28 interface. Using the **noescape** option with the autocommand feature blocks users from getting into EXEC mode.

The default X.28 router prompt is an asterisk (*). After you see *, the standard X.28 user interface is available. You configure the PAD in this mode.

To enter X.28 mode and set different access and display parameters, use the following commands in EXEC mode:

Command	Purpose
`Router> x28 escape character-string`	Specifies a character string to use to exit X.28 mode and return to EXEC mode. This string becomes an added command to X.28 mode that, when entered by the user, terminates X.28 mode and returns to EXEC mode. The default escape string is **exit**.[1]
`Router> x28 nuicud`	Places the data entered in the network user identification (NUI) facility by the user into the Call User Data (CUD) field of the X.25 call request packet.[2]
`Router> x28 profile file-name`	Specifies a user-defined X.3 profile. If this option is specified, with a profile name, then the profile is used as the initial set of X.3 parameters.[3]
`Router> x28 reverse`	Reverses the charges of all calls dialed by the local router. The address of the destination device is charged for the call. This is the default configuration. Every call is placed with the reverse charge request set.
`Router> x28 verbose`	Displays detailed information about the X.25 call connection (for example, address of the remote DTE device and the facility block used).

1. If the **x28 noescape** command is set, then it is impossible to return to the EXEC mode from X.28 mode. Use with caution. This command is not accepted when using the console line.
2. Upon entry of the **x28 nuicud** command, the network user (NU) data will not be placed in the NUI facility of the call request. Instead it will be placed in the CUD field. If you configure the **x28 nuicud** command, all reverse charging requests set by the **x28 reverse** command are disabled.
3. Profiles are created with the **x29 profile** EXEC command. If the X.29 profile is set to **default**, the profile is applied to all incoming X.25 PAD calls, including the calls used for protocol translation.

See the section "PAD Mode Connection Examples" for examples of how the **x28** and **pad** commands work.

Exchanging PAD Command Signals

The Cisco IOS universal X.28 emulation mode allows you to interact with and control the PAD. During an exchange of control information, messages or commands sent from the terminal to the PAD are called PAD command signals. Messages sent from the PAD to the terminal are called PAD service signals.

Many X.25-related functions can be performed in X.28 mode by exchanging PAD signals, such as placing and clearing calls. Table 2 lists the PAD X.28 command signals supported in the Cisco universal X.28 emulation mode.

Table 2 Available PAD Command Signals

Command	Extended Command	Purpose
break	—	Simulates an asynchronous break.
call	—	Places a virtual call to a remote device.
command-signal	—	Specifies a call request without using a standard X.28 command, which is entered with the following syntax: *facilities*-*x121-address*D*call-user-data*. The hyphen (-) and "**D**" are required keywords.
clr	**clear**	Clears a virtual call.
help	—	Displays help information.
iclr	**iclear**	Requests the remote device to clear the call.
int	**interrupt**	Sends an Interrupt packet.
par? **par**	**parameter read**	Displays the current values of local parameters.
prof	**profile** *file-name*	Loads a standard or named profile.
reset	—	Resets the call.
rpar?	**rread**	Displays the current values of remote parameters.
rset?	**rsetread**	Sets and then reads values of remote parameters.
set	—	Changes the values of local parameters. (See the "Customizing X.3 Parameters" section later in this chapter.)
set?	**setread**	Changes and then reads the values of parameters.
stat	**status**	Requests status of a connection.
selection pad	—	Sets up a virtual call.

Note You can choose to use the standard or extended command syntax. For example, you can enter the **clr** command or **clear** command to clear a call. A command specified with standard command syntax is merely an abbreviated version of the extended syntax version. Both syntaxes function the same.

Placing a Call

To place a call to another X.25 destination, you specify the destination X.121 address optionally preceded by facility requests and optionally followed by CUD. As of Cisco IOS Release 12.0, Cisco only supports the reverse charge and NUI facilities.

To place a call, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`Router> x28`	Enters X.28 mode. An asterisk prompt appears.
Step 2	`* call address`	Dials the address of the remote interface.

Note In X.28 mode, you can perform the same functions as those available with the Cisco **pad** EXEC user interface. However, X.28 mode adds functionality such as setting X.3 PAD parameters with industry-standard X.28 commands.

Clearing a Call

To clear a connection after you connect to a remote X.25 device, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`* Ctrl-p`	From the remote host, escapes back to the local router.
Step 2	`Router> clr`	Clears the virtual call.

Customizing X.3 Parameters

To set an X.3 PAD parameter from a local terminal, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`Router> x28`	Enters X.28 mode.
Step 2	`* par`	Displays the current X.3 PAD parameters.
Step 3	`* set parameter-number: new-value`	Changes the value of a parameter.
Step 4	`* par`	Verifies that the new PAD parameter was set correctly.

See Table 1 and the“X.3 PAD Parameters” appendix at the end of this publication for more information.

Accepting Reverse or Bidirectional X.25 Connections

Active lines operating in X.28 mode can receive incoming calls from the network, if they do not already have an active call. The user is notified of the call by the X.28 incoming call service signal. This feature extends the traditional capability of reverse PAD connections, which could only be received on lines that were not active.

The criteria to choose the line the call is intended for are the same as for reverse PAD connections. (The rotary is chosen from the subaddress portion of the destination address.) Because the normal rotary selection mechanism (which checks whether lines have an active EXEC) takes precedence, reverse connections to lines in X.28 mode only will work reliably to rotaries consisting of a single line.

Setting PAD French Language Service Signals

Extended dialog mode for PAD service signals is available in both the French and English languages with the PAD French Enhancement feature. The French language service signals are maintained in a table. When configured for the French language via PAD parameter 6, the PAD service signals map to

this table, giving the appropriate French equivalent output. The internal table maintenance is based on the contents of the Annex-C/X.28 standard. Section 3.5/X.28 outlines parameter 6 and how it relates to extended mode dialog in multiple languages.

The French language service signals are maintained in a table. When set for the French language via PAD parameter 6, the PAD service signals map to the French language service signals and provide the appropriate French equivalent output.

In X.28 Mode

To set French language service signals in X.28 mode, use the following commands beginning in EXEC mode:

	Command	Purpose
Step 1	`Router> x28`	Enters X.28 mode.
Step 2	`* set 6:9`	Sets the value of parameter 6 to 9 for French recognition.

Using an X.29 Profile

You can create an X.29 profile script that sets X.3 PAD parameters by using the **x29 profile** command. See the section "Creating an X.29 Profile Script" in the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" for more information about X.29 profiles.

To set French language service signals using an X.29 profile, use the following command in global configuration mode:

Command	Purpose
`Router(config)# x29 profile` *profilename* `6:9`	Sets the value of parameter 6 to 9 (on a defined set of X.3 parameters) for French recognition in an X.29 profile.

Verifying PAD French Enhancement

To verify that PAD French enhancement has been configured, enter the **parameter** command in X.28 EXEC mode (for either X.28 or X.29 profiles):

```
* parameter
   PAR 1:1 2:1 3:16 4:0 5:1 6:9 7:2 8:0 9:1 10:0 11:4 12:1 13:0 14:0 15:0 16:12 17:2 18:0
   19:0 20:0 21:0 22:0
```

Remote Access to X.28 Mode

Several ways to access X.28 PAD mode on the router are described in the following sections:

- Using an Asynchronous Line
- Using Incoming Telnet
- Using Incoming X.25

Using an Asynchronous Line

If an asynchronous line is configured with the **autocommand x28** command, the devices connected to the asynchronous line always get X.28 mode. Otherwise, an EXEC session is on the line and the **x28** command can be issued to start X.28 mode.

To set up X.28 mode on the router, perform the following the steps:

Step 1 Enter global configuration mode:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
```

Step 2 Bring up a one or more asynchronous lines and enter the **autocommand x28** command:

```
Router(config)# line 1 2
Router(config-line)# autocommand x28
```

Using Incoming Telnet

An incoming Telnet connection originates from a TCP/IP network. This connection method is used for a two-step connection from an IP device to an X.25 device.

To set up an incoming Telnet connection on the router, perform the following the steps:

Step 1 Telnet to the PAD facility inside the router.

Step 2 Instruct the PAD to connect to the X.25 device by configuring a range of virtual terminal lines to contain the **autocommand x28** command and the **rotary** *number* command:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# line vty 0 4
Router(config-line)# autocommand x28
Router(config-line)# rotary 1
Router(config-line)# exit
Router(config)#
```

Step 3 Assign an alternate IP address to the rotary port using the **ip alias** command:

```
Router(config)# ip alias aaa.bbb.ccc.ddd 3022
```

In this example, **22** is the rotary number assigned. The field **aaa.bbb.ccc.ddd** is an additional IP address assigned to the router for X.28 PAD mode incoming calls.

Step 4 The remote user accesses X.28 mode on the router by entering the **telnet aaa.bbb.ccc.ddd** command from the IP host. If required, login options can be specified on this vty.

```
ip-host% telnet 172.19.90.18

Trying 172.19.90.18...
Connected to 172.19.90.18.
Escape character is '^]'.

User Access Verification
Username: letmein
Password: guessme
```

*

Using Incoming X.25

An incoming X.25 connection originates from an X.25 network. This connection method is an unlikely scenario because most users likely are already connected to an X.25 host. However, this configuration is useful for circumventing security restrictions.

To set up incoming X.25 connection on the router, configure a range of virtual terminal lines with the **autocommand x28** command and specify a rotary number with the **rotary** *number* command.

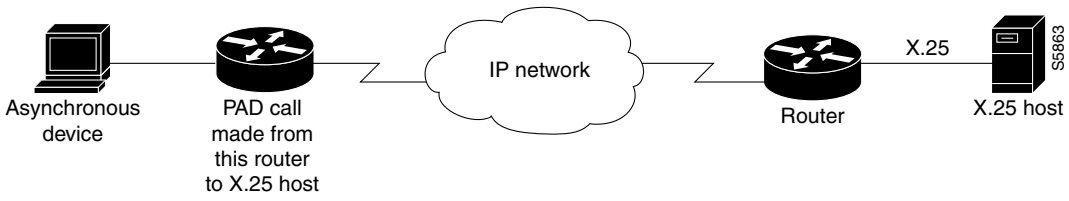
```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# line vty 0 4
Router(config-line)# autocommand x28
Router(config-line)# rotary 1
```

The remote user can now access X.28 mode by initiating a connection to the X.21 address AAAAxx, where AAAA is the X.21 address of the router and xx is the specified rotary number.

Making X.25 PAD Calls over IP Networks

PAD calls can be made to destinations that are not reachable over physical X.25 interfaces, but instead over TCP tunnels. PAD calls originating from a router on an IP link can reach an X.25 device. This feature is also known as PAD over XOT (X.25 over TCP). The **service pad to-xot** command and **service pad from-xot** global configuration command enable the PAD over XOT feature. Figure 4 shows PAD calls originating from a router in an IP network reaching an X.25 device.

Figure 4 PAD Dialing In to an X.25 Host over an IP Network



To allow PAD connections over XOT on the router, use the following commands beginning in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# configure terminal`	Enters global configuration mode.
Step 2	`Router(config)# service pad [from-xot] [to-xot]`	Specifies outgoing PAD calls over XOT or incoming XOT to PAD connections.
Step 3	`Router(config)# x25 host name x121-address` or `Router(config)# x25 route x121-address xot x121-address`	Depending on your application, specifies an X.121 address for the host name of the router or an X.25 route pointing out over XOT.[1]

1. The X.121 address of the **x25 host** command serves as a source address or sink address for PAD over XOT connections that do not have an interface. Protocol translation can also be used with incoming PAD calls over XOT, which is configured with the **translate x25** command.

Configuring PAD Subaddressing

In situations where the X.121 calling address is not sufficient to identify the source of the call, you can append a specified value to the calling address using the PAD subaddressing feature. PAD subaddressing allows you to create unique X.121 calling addresses by including either a physical port number or a value specified for a line as a subaddress to the X.121 calling address.

PAD subaddressing enables an X.25 host application to uniquely identify the source of an X.121 call. For example, in some bank security alarm applications, the central alarm host identifies the physical location of the alarm units from subaddressing information contained in the Call Request packet.

Note For an example showing PAD address substitution, see the section "Address Substitution for PAD Calls Example" in this chapter.

Before you can configure PAD subaddressing, you need to configure your router or access server to support X.25. For more information, refer to the *Cisco IOS Wide-Area Networking Configuration Guide*, Release 12.2.

To configure PAD subaddressing, use the following commands beginning in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# configure terminal`	Enters global configuration mode.
Step 2	`Router(config)# line [aux | console | tty | vty] line-number [ending-line-number]`	Identifies the line(s) whose information will be appended to the X.121 address as the subaddress.
Step 3	`Router(config-line)# x25 subaddress {line | number}`	Creates a unique X.121 calling address by adding either a physical port number or a numeric value for a line as a subaddress to the X.121 calling address.

Configuring X.29 Reselect

Cisco supports X.29 reselect, which is a standard Triple-X PAD function supported in later versions of the X.3, X.28, and X.29 specifications. X.29 reselect is used in conjunction with mnemonics and autoconnect/autocall to the "first host." X.29 reselect is for security checking and DNS, such as the X.25 naming/selection of destinations within a public or private network. The primary (first) destination host acts much like a RADIUS/TACACS server. At a minimum, both the PAD and the "first host" used in the topology need to support X.29 reselect. X.29 reselect is transparent to network elements or switches. No Cisco IOS commands need to be entered to enable X.29 reselect. It is enabled by default.

Using Mnemonic Addressing

Mnemonic addressing enables you to connect to a remote host by using its mnemonic address, not the X.121 address. As the number of hosts grows within an X.25 network, system administrators need to remember numerous 14-digit X.121 addresses to connect to multiple host applications. To ease the burden of this administrative overhead, asynchronous PAD users can now access hosts by using mnemonic (abbreviated) addressing.

When the user specifies the mnemonic address in the **call** X.28 command, the mnemonic gets translated to an X.121 address in the local PAD. The resulting call request contains both the X.121 calling and called addresses.

Note For an example showing PAD address substitution, see the section "Address Substitution for PAD Calls Example" in this chapter.

Character Limitations

You can use the following formats to specify a mnemonic address:

- Any combination of numbers, letters, and special characters preceded by a dot, or period (.)
- Up to 250 characters in one address

Note All other facilities provided in X.28 emulation mode remain the same.

Mnemonic Format Options

This section provides examples of format options.

Example 1

Format

c <NUI, Facilities>-.<Mnemonic>*<call-user-data>

Description

This is the generalized format of the **call** command where you can specify NUI and facilities with -.mnemonics and an asterisk (*) before the call user data (CUD). The comma (,) separates individual facility specifications.

Example Syntax

Nsmith-.billing*xyz

In this example, the following facilities are specified:

```
smith = NUI and no facilities
billing = 31xx4085272478
xyz = CUD
```

Example 2

Format

c .<Mnemonic>*<call-user-data>

Description

No facilities, with CUD.

Example Syntax

c .billing*xyz

In this example, the following facility is specified:

```
billing = 31xx4085272478 with CUD of xyz
```

Example 3

Format

c <Mnemonic>

Description

No dot, no facilities, no CUD.

Example syntax

billing

In this example, the following facility is specified:

```
billing = 31xx4085272478
```

Example 4

Format

<Mnemonic>

Description

No dot, no facilities, no CUD.

Example Syntax

billing

In this example, the following facility is specified:

```
billing = 31xx4085272478
```

Facility Codes

Table 3 lists the supported facility codes that can be specified in the Call Request packet. The X.121 address is a *word* with decimal digits.

Table 3 Facility Codes

Code	Description
N *word*	NUI.
T *word*	Recognized Private Operating Agency (RPOA).
R	Reverse charge.
G *word*	Closed user group (*word* is one or two decimal digits).
O *word*	Closed user group with outgoing access (*word* is one or two decimal digits).
C	Charging information.
E *word*	Called address (*word* is up to 40 decimal digits).
F	Fast select with no restrictions.
S	Reselect prevention.
Q	Fast select with restrictions.

PAD Examples

This section provides the following PAD connection and configuration examples:

- PAD EXEC User Interface Connection Examples
- Cisco Universal X.28 PAD Emulation Mode Examples
- PAD XOT Examples
- PAD Subaddressing Examples

PAD EXEC User Interface Connection Examples

This section provides the following examples of making PAD connections using the **pad** command:

- PAD Mode Connection Examples
- X.3 Parameter Customization Example
- Load an X.3 Profile Example
- Set PAD Parameters Example

PAD Mode Connection Examples

The following examples show two ways to make a call to a remote X.25 host over a serial line. The interface address of the remote host is 123456. In the first example, Router-A calls Router-B using the **pad 123456** EXEC command. The second example shows Router-A calling Router-B using the **call 123456** PAD signal command in X.28 mode. Both commands accomplish the same goal.

```
Router-A# pad 123456
Trying 123456...Open

Router-B> exit

[Connection to 123456 closed by foreign host]

Router-A# x28

* call 123456
COM

Router-B>
```

The following examples show two ways to clear a connection with a remote X.25 host. The first example shows Router-A disconnecting from Router-B using the **disconnect** command in EXEC mode. The second example shows Router-B disconnecting from Router-A using the **clr** command in X.28 mode.

```
Router-A# pad 123456
Trying 123456...Open

Router-B> <Enter the escape sequence (for example, press Shift-Ctrl-^-x).>

Router-A# disconnect
Closing connection to 123456 [confirm]
Router-A#

Router-A# x28

* call 123456
COM

Router-B> <Press Ctrl-p>
* clr

CLR CONF

*
```

X.3 Parameter Customization Example

The following example shows how to change a local X.3 PAD parameter from a remote X.25 host using X.29 messages, which is a secure way to enable a remote host to gain control of local PAD. The local device is Router-A. The remote host is Router-B. The parameters listed in the ParamsIn field are incoming parameters, which are sent by the remote PAD. The parameters listed in the ParamsOut field are parameters sent by the local PAD.

```
Router-A# pad 123456
Trying 123456...Open

Router-B> x3 2:0
Router-B>
Router-A# show x25 pad
```

```
tty0, connection 1 to host 123456

Total input: 12, control 3, bytes 35. Queued: 0 of 7 (0 bytes).
Total output: 10, control 3, bytes 64.
Flags: 1,   State: 3,   Last error: 1
 ParamsIn:  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
    8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
    16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0,
 ParamsOut:  1:1, 2:0, 3:2, 4:1, 5:1, 6:0, 7:21,
    8:0, 9:1, 10:0, 11:14, 12:1, 13:0, 14:0, 15:0,
    16:127, 17:21, 18:18, 19:0, 20:0, 21:0, 22:0,
Router-A#
```

Load an X.3 Profile Example

The following example modifies and loads an existing X.25 PAD parameter profile. It accesses the existing PAD profile ppp, changes its padding parameter (specified as 9) to a value of 2, and displays the new parameters using the **par** command in X.28 mode.

```
Router-A# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router-A(config)# x29 profile ppp 9:2
Router-A(config)# end
Router-A#
%SYS-5-CONFIG_I: Configured from console by console
Router-A# x28 profile ppp

* par
   PAR 1:1 2:1 3:126 4:0 5:1 6:2 7:2 8:0 9:2 10:0 11:14 12:1 13:0 14:0 15:0 16:127 17:24
   18:18 19:2 20:0 21:0 22:0
```

Note If the X.29 profile is set to **default**, the profile is applied to all incoming X.25 PAD calls, including the calls used for protocol translation.

Set PAD Parameters Example

The following example starts a PAD session:

```
Router> pad 123456789
Trying 123456789...Open
Router2>
```

The following example shows how to reset the outgoing connection default for local echo mode on a router. The **/set** switch sets the X.3 parameters defined by parameter number and value, separated by a colon.

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

The following are examples of **show x25 vc** command output for PAD over Connection-Mode Network Service (CMNS), PAD to PAD over X.25, and PAD over XOT (X.25 over TCP) connections:

```
Router# show x25 vc

SVC 1,  State: D1,  Interface: Ethernet0
  Started 00:01:48, last input 00:01:48, output 00:01:48

  Line: 0   con 0    Location: console Host: 2193330
   connected to 2193330 PAD <--> CMNS Ethernet0 00e0.b0e3.0d62
```

```
  Window size input: 2, output: 2
  Packet size input: 128, output: 128
  PS: 2  PR: 3  ACK: 3  Remote PR: 2  RCNT: 0  RNR: no
  P/D state timeouts: 0  timer (secs): 0
  data bytes 54/19 packets 2/3 Resets 0/0 RNRs 0/0 REJs 0/0 INTs 0/0

SVC 1024,  State: D1,  Interface: Serial1
  Started 00:00:07, last input 00:00:26, output 00:00:26

  Line: 0   con 0    Location: console Host: 2194443
  2191111 connected to 2194443 PAD <--> X25

  Window size input: 5, output: 5
  Packet size input: 128, output: 128
  PS: 0  PR: 0  ACK: 0  Remote PR: 0  RCNT: 0  RNR: no
  P/D state timeouts: 0  timer (secs): 0
  data bytes 0/0 packets 0/0 Resets 0/0 RNRs 0/0 REJs 0/0 INTs 0/0

SVC 1,  State: D1,  Interface: [172.21.9.7,1998/172.21.9.11,11000]
  Started 00:06:48, last input 00:06:43, output 00:06:43

  Line: 0   con 0    Location: console Host: 219444001
  219111 connected to 219444001 PAD <--> XOT 172.21.9.7,1998

  Window size input: 2, output: 2
  Packet size input: 128, output: 128
  PS: 5  PR: 4  ACK: 4  Remote PR: 5  RCNT: 0  RNR: no
  P/D state timeouts: 0  timer (secs): 0
  data bytes
```

The following example shows output for the **show x25 pad** command:

```
Router# show x25 pad

tty0 (console), connection 1 to host 2194440

Total input: 75, control 2, bytes 3168. Input Queued: 0 of 7 (0 bytes).
Total output: 50, control 2, bytes 52. Output Queued: 0 of 5.
Flags: 1,   State: 3,   Last error: 1
 ParamsIn:  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
    8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
    16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0,
 ParamsOut:  1:1, 2:0, 3:2, 4:1, 5:1, 6:0, 7:21,
    8:0, 9:0, 10:0, 11:14, 12:1, 13:0, 14:0, 15:0,
    16:127, 17:21, 18:18, 19:0, 20:0, 21:0, 22:0,

tty18, Incoming PAD connection
Total input: 2, control 2, bytes 54. Input Queued: 0 of 7 (0 bytes).
Total output: 1, control 2, bytes 9. Output Queued: 0 of 5.
Flags: 1,   State: 3,   Last error: 1
 ParamsIn:  1:1, 2:0, 3:2, 4:1, 5:0, 6:0, 7:21,
    8:0, 9:0, 10:0, 11:14, 12:0, 13:0, 14:0, 15:0,
    16:127, 17:21, 18:18, 19:0, 20:0, 21:0, 22:0,
 ParamsOut:  1:1, 2:1, 3:2, 4:1, 5:0, 6:0, 7:4,
    8:0, 9:0, 10:0, 11:14, 12:0, 13:0, 14:0, 15:0,
    16:127, 17:21, 18:18, 19:0, 20:0, 21:0, 22:0,
```

Cisco Universal X.28 PAD Emulation Mode Examples

This section contains the following examples of making PAD connections using the **x28** command:

Set Parameters Using X.28 PAD Emulation Mode Example

The following example configures parameter 9 from 0 to 1, which adds a byte after the carriage return. This setting is performed from a local terminal using the **set** *parameter-number*:*new-value* PAD command signal.

```
Router# x28

* par
   PAR 1:1 2:1 3:126 4:0 5:1 6:2 7:2 8:0 9:0 10:0 11:14 12:1 13:0 14:0 15:0 16:127 17:24
   18:18 19:2 20:0 21:0 22:0

* set 9:1

* par
   PAR 1:1 2:1 3:126 4:0 5:1 6:2 7:2 8:0 9:1 10:0 11:14 12:1 13:0 14:0 15:0 16:127 17:24
   18:18 19:2 20:0 21:0 22:0

*
```

NUI Data Relocation Example

The following example sends an authentication message to a remote X.25 host using the **x28 nuicud** command in Cisco X.28 mode followed by the **Ncisc-123456** command. The network identifier is N. The network user password is cisc. The destination address of the remote device is 123456. The ASCII representation of the user password appears in the CUD field, not in the data packet.

```
Router-A# debug x25 event
X.25 special event debugging is on
Router-A# x28 nuicud

* Ncisc-123456
COM

Router-B>
02:02:58: Serial1: X.25 O P1 Call (16) 8 lci 20
02:02:58:   From(3): 222 To(3): 123456
02:02:58:   Facilities: (0)
02:02:58:   Call User Data (8): 0x01000000xxxxxxxx (pad)
02:02:58: Serial1: X.25 I P2 Call Confirm (5) 8 lci 20
02:02:58:   From(0):  To(0):
02:02:58:   Facilities: (0)
```

X.25 Reverse Charge Example

The following example shows how to use the **x28 reverse** command to make the charges for all outgoing calls made from the local router be reversed to the destination device. To reverse the charges for only one outgoing call, use the **R-***address* command, which is the standard X.28 reverse charge facility command.

```
Router-A# x28 reverse

* exit

Router-A# x28

* R-123456
COM
```

X.25 Call Detail Display Example

Each time a call is made to a remote device, you can specify that detailed information be displayed about the call and the destination device by entering the **x28 verbose** command. The following example shows reverse charging configured and CUD represented as userdata:

```
Router# x28 verbose

* R-111*userdata

Called DTE Address : 3001
Facility Block     : R
Call User Data     :userdata
COM
```

Set PAD French Service Signals in X.28 Mode Example

The following example shows PAD French enhancement being set in X.28 EXEC mode:

```
Router # x28
* set 6:9
```

Set PAD French Service Signals with an X.29 Profile Example

The following example shows PAD French enhancement being set with an X.29 profile:

```
Router(config)# x29 profile Primary 6:9
```

Get Help Example

The following example shows how to use the **help** command to get short descriptions of the available parameters:

```
* help
The "help" PAD command signal consists of the following elements:
 <help PAD command signal> <help subject>
  where
 <help subject> is the identifier for the type of
                explanatory information requested
* help break
BREAK    Simulate async BREAK
```

PAD XOT Examples

The following sections provide PAD over XOT configuration examples:

Accept XOT to PAD Connections Example

The following example enables connections from XOT to a local PAD. Because XOT is a TCP connection, the connection is not tied to an X.25 interface. An X.25 address must be configured for the host name of the router that is accepting the call. In this case, the router answers and clears an incoming PAD call through address 1234.

```
Router(config)# service pad from-xot
Router(config)# x25 host Router-A 1234
```

Accept XOT to Protocol Translation Example

The following example accepts an incoming PAD call over XOT to address 12345. The router then translates the call and makes a TCP connection to the device named puli.

```
Router(config)# service pad from-xot
Router(config)# translate x25 12345 tcp puli
```

Initiate a PAD Call over an XOT Connection Example

The following example enables outgoing PAD to XOT connections from an asynchronous line or vty. A route pointing out over XOT must be configured on the routing table to make a PAD call. This route can also be used for switching.

```
Router(config)# service pad to-xot
Router(config)# x25 route 1111 xot 10.2.2.2.
```

Address Substitution for PAD Calls Example

X25 synchronous or PAD devices attached to a router in a remote location may need to ensure that outgoing PAD calls use an assigned X.121 address for the calling (source) address or an assigned X.121 address for the called (destination) address.

Normally, the called address is sent by default in the outgoing PAD call. For the source address, the PAD applies the address for the originating interface (even if it is NULL) or the X25 host address (for example, XOT) as the source address of the call. To override the default behavior and substitute the original X.121 source/destination address in the outgoing PAD calls, use the **x25 route** command with the **substitute-source** and **substitute-dest** keyword options.

Note Address substitution can be applied to all PAD connections, not just PAD over XOT.

Configuring Address Substitution

The following example performs address substitution for PAD calls over XOT:

```
Router(config)# x25 route ^1234 substitute-source 5678 xot 10.1.1.1
```

or

```
Router(config)# x25 route ^1234 substitute-dest 5678 interface serial 1
```

Verifying Address Substitution

To verify the source or destination address substitution on the outgoing PAD call, use the **debug x25 event** command and **show x25 vc** command.

For example, to substitute the destination address of 8888 to 5678 and replace the default source address of the outgoing PAD call to 1234, enter the following **x25 route** command:

```
Router(config)# x25 route 8888 substitute-source 1234 substitute-dest 5678 interface
serial 1
```

Placing a PAD call to destination 8888 will be substituted by 5678 and a source address of 1234:

```
Router# pad 8888

Trying 8888...Open
```

The following is output of the **x25 debug event** command:

```
Serial1: X.25 O R1 Call (13) 8 lci 1024
   From(4): 1234 To(4): 5678
   Facilities: (0)
   Call User Data (4): 0x01000000 (pad)
Serial1: X.25 I R1 Call Confirm (5) 8 lci 1024
   From(0):  To(0):
   Facilities: (0)
```

The following is output from the **show x25 vc** command:

```
Router# show x25 vc

SVC 1024,  State: D1,  Interface: Serial1
  Started 00:23:54, last input 00:00:13, output 00:00:13

  Line: 0   con 0    Location: console Host: 456
  1234 connected to 5678 PAD <--> X25

  Window size input: 2, output: 2
  Packet size input: 128, output: 128
  PS: 0  PR: 0  ACK: 0  Remote PR: 0  RCNT: 0  RNR: no
  P/D state timeouts: 0  timer (secs): 0
  data bytes 68/958 packets 16/27 Resets 0/0 RNRs 0/0 REJs 0/0 INTs 0/0
```

PAD Subaddressing Examples

The following example shows how to configure subaddressing on virtual terminal lines 10 through 20 by appending the line number as a subaddress to the X.121 calling address:

```
Router(config)# line vty 10 20
Router(config-line)# x25 subaddress line
```

The following example shows how to configure subaddressing on the first five TTY lines by appending the value 9 as a subaddress to the X.121 calling address of the X.28 connection originating on these lines:

```
Router(config-line)# line 1 5
Router(config-line)# x25 subaddress 9
Router(config-line)# autocommand x28
```

You can use the output from the **debug x25 event** and the **show line** commands to display information about PAD subaddressing. Once you have configured PAD subaddressing, the output from both of these commands changes to reflect the additional subaddress information.

The following example shows **debug x25 event** output, where the X.25 address is 12345 and the subaddress for TTY line 3 is 09:

```
Router# debug x25 event

Serial1: X.25 O P1 Call (14) 8 lci 1024
  From(7): 1234509 To(4): 6789
  Facilities: (0)
  Call User Data (4): 0x01000000 (pad)
Serial1: X.25 I P2 Call Confirm (5) 8 lci 1024
  From (0): to (0):
  Facilities: (0)
  PAD3: Call completed
```

The following example shows sample **show line** output for a router named enkidu, where line 18 has been configured for PAD subaddressing:

```
Router# show line 18

Tty  Typ     Tx/Rx     A Modem  Roty  AccO  AccI  Uses  Noise  Overruns
 18  VTY               -    -      -     -     -     1      0       0/0

Line 18, Location: "enkidu", Type: " "
Length: 48 lines, Width: 80 columns
Baud rate: (TX/RX) is 9600/9600
Status: Ready, Connected, Active, No Exit Banner
Capabilities: Line usable as async interface, PAD Sub-addressing used
Modem state: Ready
```

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

PAD Subaddress Formatting Option

Prior to Cisco IOS Release 12.3(2)T, packet assembler/disassembler (PAD) Subaddressing specifies a two-digit field for subaddressing that requires a leading zero for subaddress values of nine or lower (0-9). The PAD Subaddress Formatting Option feature introduces the ability to suppress the leading zero for subaddresses with a value of nine or lower. This suppression occurs before the subaddress field is appended to the calling address. This feature increases compatibility with X.25 host systems that use single-digit subaddresses.

Feature History for the PAD Subaddress Formatting Option Feature

Release	Modification
Cisco IOS XE Gibraltar 16.10.1	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

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

Prerequisites for PAD Subaddress Formatting Option

PAD must be configured. For more information on configuring PAD, refer to Configuring the Cisco PAD Facility for X.25 Connections.

Restrictions for PAD Subaddress Formatting Option

X.25 subaddresses in the range from 1 to 99 are tied to rotary groups and can be only two digits in length.

The PAD Subaddress Formatting Option feature is available for the following line types:

- CON
- AUX
- TTY
- VTY

The PAD Subaddress Formatting Option feature is supported for the following connection types:

- PAD
- X28
- PT

Information About PAD Subaddress Formatting Option

To configure the PAD Subaddress Formatting Option feature, you must understand the following concepts:

- PAD Subaddress Values, page 2
- Benefits of the PAD Subaddress Formatting Option, page 2

PAD Subaddress Values

PAD subaddressing enables an X.25 host application to uniquely identify the source of an X.121 call. In some situations, the X.121 calling address alone is not sufficient to identify the source of the call. PAD subaddressing allows you to create unique X.121 calling addresses by including either a physical port number or an explicit value to be specified for a line as a subaddress to the X.121 calling address.

The PAD Subaddress Formatting Option feature introduces the option to exclude the leading zero from PAD subaddress with a value of nine or lower (0-9). This option affects only the formatting of the PAD subaddress, not the value of the PAD subaddress. The PAD subaddress 02 has exactly the same value as the PAD subaddress 2.

A single Cisco router can be configured to generate PAD subaddresses with and without leading zeros on different lines or sets of lines.

Benefits of the PAD Subaddress Formatting Option

This feature increases compatibility with X.25 host systems that use single-digit subaddresses.

How to Configure PAD Subaddress Formatting Option

This section contains the following procedure:

- Configuring the PAD Subaddress Formatting Option, page 3 (required)

Configuring the PAD Subaddress Formatting Option

This task configures a set of lines to suppress the leading zero for subaddresses with a value of nine or lower (0-9).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*]
4. **x25 subaddress** {**line** | *number*} [**no-zero-pad**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*] **Example:** `Router(config)# line vty 1 9`	Enters line configuration mode and identifies a specific line or set of lines for configuration.
Step 4	**x25 subaddress** {**line** | *number*} [**no-zero-pad**] **Example:** `Router(config-line)# x25 subaddress 6 no-zero-pad`	Appends either a physical port number or a value specified for a line as a subaddress to the X.121 calling address. • **no-zero-pad**—Specifies that a leading zero should not be appended to subaddresses with a value of nine or lower (0-9).

Configuration Examples for PAD Subaddress Formatting Option

This section contains the following configuration examples:

Configuring the PAD Subaddress Formatting Option Example

The following example configures a subaddress of 6 for a set of vty lines, and specifies that a leading zero should not be appended to the subaddress value:

```
Router(config)# line vty 0 9
Router(config-line)# x25 subaddress 6 no-zero-pad
```

Verifying Configuration of the PAD Subaddress Formatting Option Example

To verify the configuration of the PAD Subaddress Formatting Option, enter the **show line** command as shown in the following example:

```
Router# show line vty 0

Tty Typ Tx/Rx A Modem Roty AccO AccI Uses Noise Overruns Int
66 VTY - - - - - 0 0 0/0 -
Line 66, Location: "", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600
Status: No Exit Banner
Capabilities: PAD Sub-addressing Used, No Leading Zeros
Modem state: Idle
Group codes: 0
Special Chars: Escape Hold Stop Start Disconnect Activation
^^x none - - none
Timeouts: Idle EXEC Idle Session Modem Answer Session Dispatch
never never none not set
Idle Session Disconnect Warning
never
Login-sequence User Response
00:00:30
Autoselect Initial Wait
not set
Modem type is unknown.
```

Additional References

The following sections contain additional information related to the PAD Subaddress Formatting Option feature.

Related Documents

Related Topic	Document Title
Information on configuring PAD	*Configuring the Cisco PAD Facility for X.25 Connections*
Additional PAD commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Terminal Services Command Reference*

Standards

Standards	Title
None	—

MIBs

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

RFCs

RFCs	Title
None	—

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

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

- **x25 subaddress**

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved.

Configuring Protocol Translation and Virtual Asynchronous Devices

This chapter describes how to configure protocol translation and virtual asynchronous connections using Cisco IOS software. The tasks are described in the following sections, which also describe the process of tunneling and protocol translation, and the two-step and one-step translation methods:

The X.3 packet assembler/disassembler (PAD) parameters are described in the "X.3 PAD Parameters" appendix later in this chapter.

The protocol translation facility assumes that you understand how to use the configuration software. Before using this chapter, you should be familiar with configuring the protocols for which you want to translate: X.25, Telnet, local-area transport (LAT), TN3270, AppleTalk Remote Access (ARA), PPP, Serial Line Internet Protocol (SLIP), and XRemote.

Note Telnet is a remote terminal protocol that is part of the TCP/IP suite. The descriptions and examples in the following sections use the term TCP as a reference to the Telnet functionality.

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

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the "Using Cisco IOS Software" chapter.

For a complete description of the commands in this chapter, refer to *Cisco IOS Terminal Services Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

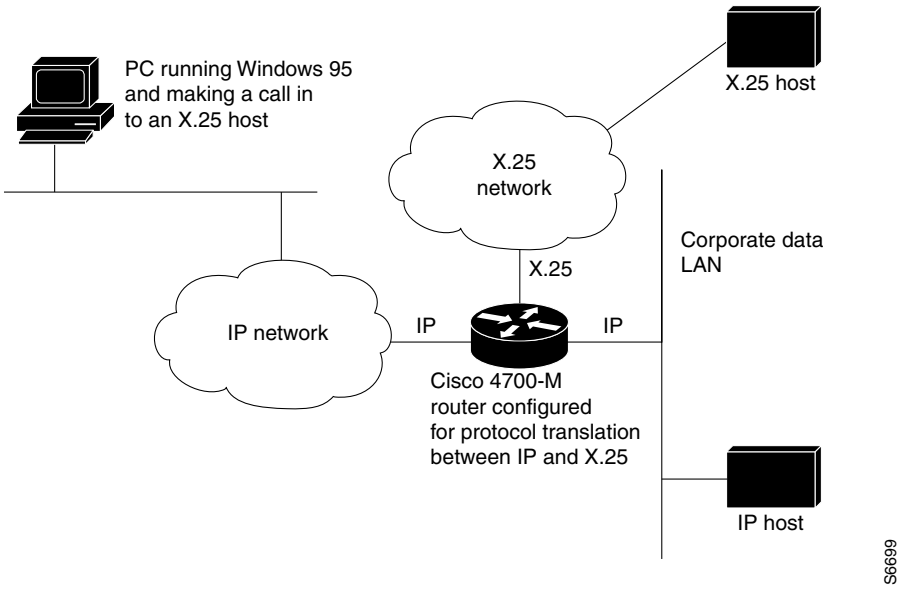
Protocol Translation Overview

Definition of Protocol Translation

The protocol translation feature provides transparent protocol translation between systems running different protocols. It enables terminal users on one network to access hosts on another network, despite differences in the native protocol stacks associated with the originating device and targeted host.

Protocol translation is a resourceful facility for many business applications. For example, Figure 1 shows a remote PC dialing through an IP network and connecting to an X.25 host. The TCP packets on the PC undergo a TCP-to-X.25 protocol translation by the Cisco 4700-M router.

Figure 1 Protocol Translation Business Application

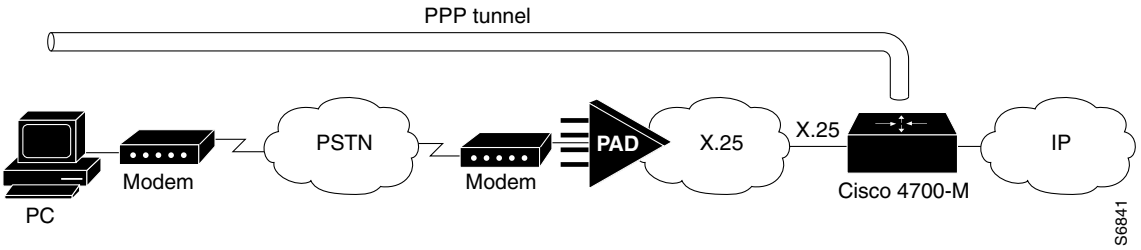


Definition of Tunneling

Unlike other protocols such as LAT, X.25, and TCP, which are actually translated when you use protocol translation, SLIP, PPP, and ARA are not translated to the destination protocol. Instead, they are carried inside a LAT, X.25, TCP, or Layer 2 Forwarding Protocol (L2F) tunnel specific to the device on the remote network. However, the protocol translation facility is used to enable tunneling of SLIP, PPP, or ARA.

Figure 2 shows a typical tunneling scenario.

Figure 2 Tunneling X.25 with PPP Across an IP Network



You can also tunnel PPP-IPX over X.25, TCP, or LAT to an Internetwork Packet Exchange (IPX) network when tunneling PPP on virtual terminal lines.

Deciding Whether to Use One-Step or Two-Step Protocol Translation

Cisco IOS software supports virtual terminal connections in both directions between the following protocols. You can configure the router to translate automatically between them. This translation method is called *one-step translation*, and is more popular than the two-step method.

- X.25 and LAT
- X.25 and Telnet sessions using the TCP
- LAT and TCP/Telnet

On outgoing connections, you can also use the one-step protocol translation facility to tunnel SLIP or PPP to IP and IPX networks, or ARA to AppleTalk networks across X.25, LAT, or IP (on outgoing connections only).

Cisco IOS software supports limited connections in both directions between the following protocols. Connecting between these protocols requires that you first connect to a router, and then to the host to which you want to connect. This translation method is called *two-step translation*, and is the less popular method.

- XRemote to SLIP/PPP and X.25 PAD environments (XRemote must use the two-step method)
- LAT, X.25, SLIP/PPP, and TCP (Telnet) to TN3270 (TN3270 must use the two-step method)

One-Step Protocol Translation

Use the one-step method when network users repeatedly log in to the same remote network hosts through a router. This connection is more efficient than the two-step method and enables the device to have more knowledge of the protocols in use because the router acts as a network connection rather than as a

terminal. The one-step method provides transparent protocol conversion. When connecting to the remote network host, the user enters the connection command to the remote network host but does not need to specify protocol translation. The network administrator has already created a configuration that defines a connection and the protocols to be translated. The user performs only one step to connect with the host.

When you make a one-step connection to the router, the Cisco IOS software determines the host for the connection and the protocol the host is using. It then establishes a new network connection using the protocol required by that host.

A disadvantage of the one-step protocol translation method is that the initiating computer or user does not know that two networking protocols are being used. This limitation means that parameters of foreign network protocols cannot be changed after connections are established. The exception to this limitation is any set of parameters common to both networking protocols; any parameter common to both can be changed from the first host to the final destination.

To configure the one-step method of protocol translation, set up the following protocols and connection options in the configuration file:

- The incoming connection—The configuration includes the protocol to be used—LAT, X.25, or TCP/IP (Telnet)—the address, and any options such as reverse charging or binary mode that are supported for the incoming connection.
- The outgoing connection—The outgoing connection is defined in the same way as the incoming connection, except that SLIP, PPP (including IP and IPX on PPP sessions), and ARA are also supported.
- The connection features global options—You can specify additional features for the connection to allow, for example, incoming call addresses to match access list conditions or limit the number of users that can make the connection.

Refer to the "Protocol Translation Configuration Task List" section on page 5 for configuration tasks.

Two-Step Protocol Translation

Use two-step protocol translation for one-time connections or when you use the router as a general-purpose gateway between two types of networks (for example, X.25 public data network (PDN) and TCP/IP). As with the one-step method, it is recommended that you configure virtual templates for this feature.

Note Use the two-step method for translations of TN3270 and XRemote.

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

Refer to the "Protocol Translation Configuration Task List" section on page 5 for configuration tasks.

Tunneling SLIP, PPP, and ARA

Unlike other protocols such as LAT, X.25, and TCP, which actually are translated when you use one-step protocol translation, SLIP, PPP, and ARA are not translated to the destination protocol. Instead, they are carried inside a LAT, X.25, or TCP tunnel specific to the device on the remote network. However, you can use the protocol translation facility to enable tunneling of SLIP, PPP, or ARA.

You can also tunnel IPX-PPP over X.25, TCP, or LAT, to an IPX network when tunneling PPP on virtual terminal lines. Refer to the "Configuring X.29 Access Lists" section on page 6 for configuration tasks.

Protocol Translation Configuration Task List

Configuring One-Step Protocol Translation

To create one-step protocol translation connection specifications, use the following command in global configuration mode:

Command	Purpose
`Router(config)# translate protocol incoming-address`	Creates the connection specifications for one-step protocol translations.

For incoming PAD connections, the router uses a default PAD profile to set the remote X.3 PAD parameters unless a profile script is defined in the **translate** command. To override the default PAD profile that the router uses, you must create a PAD profile script using the **x29 profile** global configuration command. In the following example, *default* is the name of the default PAD profile script and *parameter:value* is the X.3 PAD parameter number and value separated by a colon.

```
x29 profile default parameter:value [parameter:value]
```

Note If the X.29 profile is named default, it is applied to all incoming X.25 PAD calls, including the calls used with protocol translation.

Configuring a Virtual Template for Two-Step Protocol Translation

If you are tunneling PPP or SLIP using two-step protocol translation with virtual interface templates, you will still use the **vty-async** command before implementing virtual templates. However, virtual asynchronous interfaces are created dynamically when a tunnel connection is established.

To create and configure a virtual interface template and apply it to a two-step protocol translation session, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface virtual-template number`	Creates a virtual interface template, and enters interface configuration mode.
Step 2	`Router(config-if)# ip unnumbered ethernet 0`[1]	Assigns an IP address to the virtual interface template.
Step 3	`Router(config-if)# encapsulation {ppp | slip}`[2]	Enables encapsulation on the virtual interface template.
Step 4	`Router(config-if)# peer default ip address {dhcp | pool [pool-name-list]}`	Assigns an IP address from a pool to the device connecting to the virtual access interface (such as the PC in Figure 3).
Step 5	`Router(config-if)# exit`	Returns to global configuration mode.
Step 6	`Router(config)# vty-async`	Creates a virtual asynchronous interface.
Step 7	`Router(config)# vty-async virtual-template number`	Applies the virtual template to the virtual asynchronous interface.

1. You can also assign a specific IP address by using the **ip address** *address* command, though assigning the IP address of the Ethernet0 interface as shown is most common.
2. Virtual interface templates use PPP encapsulation by default, so you need not specify **encapsulation ppp**. However, to use SLIP encapsulation, you must explicitly specify **encapsulation slip**.

Other asynchronous configuration commands can be added to the virtual template configuration. For example, you can enter the **ppp authentication chap** command. It is recommended that you include security on your virtual interface template.

Configuring X.29 Access Lists

Cisco IOS software provides access lists to limit access to a router from certain X.25 hosts. Access lists take advantage of the message field defined by Recommendation X.29, which describes procedures for exchanging data between two PADs or between a PAD and a DTE device.

To define X.29 access lists, perform the tasks described in these sections:

- Creating an X.29 Access List, page 6 (required)
- Applying an Access List to a Virtual Line, page 7 (required)

Note When configuring protocol translation, you can specify an access list number with each **translate** command. In the case of translation sessions that result from incoming PAD connections, the corresponding X.29 access list is used.

Creating an X.29 Access List

To specify the access conditions, use the following command in global configuration mode:

Command	Purpose
`Router(config)# x29 access-list access-list-number {permit | deny} regular-expression`	Restricts incoming and outgoing connections between a particular vty (into a router) and the addresses in an access list.

An access list can contain any number of lines. The lists are processed in the order in which you type the entries. The first match causes the permit or deny condition. If an X.121 address does not match any of the entries in the access list, access will be denied.

Applying an Access List to a Virtual Line

To apply an access list to a virtual line, use the following command in line configuration mode:

Command	Purpose
`Router(config-line)# access-class number in`	Restricts incoming and outgoing connections between a particular vty (into a router) and the addresses in an access list.

The access list number is used for incoming TCP and PAD accesses. For TCP access, the access server or router using protocol translation uses the defined IP access lists. For incoming PAD connections, the same X.29 access list is used. If you want to apply access restrictions on only one of the protocols, create an access list that permits all addresses for the other protocol.

Note For an example of including an access list in a **translate** command.

Changing the Number of Supported Translation Sessions

There is a one-to-one relationship between protocol translation sessions and virtual terminal lines. For every session, you need a vty. Therefore, if you need to increase the number of protocol translation sessions, you need to increase the number of virtual terminal lines. That is, if your router has ten virtual terminal lines, you can have ten protocol translation sessions. The default number of virtual terminal lines is 5 (lines 0 through 4).

To increase the number of lines and correspondingly increase the number of protocol translation sessions, use the following commands in global configuration mode:

Command	Purpose
`Router(config)# line vty line-number`	Increases the number of virtual terminal lines.
`Router(config-line)# no line vty line-number`	Decreases the number of virtual terminal lines.

Protocol translation is a CPU-intensive task. Increasing the number of protocol translation sessions while routing is enabled can impact the available memory. The amount of memory available depends on the platform type, the amount of DRAM available, the activity of each translation session, and the speed of the link. If you are using the maximum number of sessions and have problems with memory, you might need to decrease the number of protocol translation sessions.

Creating an X.29 Profile Script

You can create an X.29 profile script for the **translate** command to use. An X.29 profile script uses X.3 PAD parameters. When an X.25 connection is established, Cisco IOS software configured for protocol translation functions similar to an X.29 SET PARAMETER packet, which contains the parameters and values set by this command.

To create an X.29 profile script, use the following command in global configuration mode:

Command	Purpose
`Router(config)# x29 profile {default | name} parameter:value [parameter:value]`	Creates an X.29 profile script.

For incoming PAD connections, the router running protocol translation uses a default PAD profile to set the remote X.3 PAD parameters, unless a profile script is defined in the **translate** command. To override the default PAD profile that the router uses, you must create a PAD profile script and name it default using the **x29 profile** {**default** | *name*} *parameter:value* [*parameter:value*] global configuration command, where the *name* argument is the word "default" and *parameter:value* is the X.3 PAD parameter number and value separated by a colon. For more information about X.3 PAD parameters, refer to the appendix "X.3 PAD Parameters" at the end of this publication.

Note When the X.29 profile is named default, it is applied to all incoming X.25 PAD calls, including the calls used with protocol translation.

You can also create an X.29 profile script when connecting to a PAD using the **pad** [**/profile** *name*] EXEC command, which is described in *Cisco IOS Terminal Services Command Reference*.

Defining X.25 Hostnames

This section describes how to define symbolic hostnames, which means that instead of remembering a long numeric address for an X.25 host, you can refer to the X.25 host using a symbolic hostname. To define a symbolic hostname, use the following command in global configuration mode:

Command	Purpose
`Router(config)# x25 host name x.121-address [cud call-user-data]`	Defines a symbolic hostname.

Protocol Translation and Processing PAD Calls

Background Definitions and Terms

X.29 encodes the PAD Call User Data (CUD) field in the call packet to indicate that the call request signifies a PAD-to-DTE device interaction.The CUD field is 16 bytes long and can be up to 128 bytes long when the "Select" facility is applied. The first 4 bytes of the CUD field represent the protocol identifier (PID).

When a PAD calls a host DTE device, X.29 ensures that the encoding of the PID field contains a standard PAD PID "0x01000000," which informs the host that a PAD is calling. The remainder of the CUD field contains the user data that could signify a login message or a password for the host.

The **x25 map pad** interface command specifies the other end of a connection and how to interact with that host. For incoming calls, the PAD checks for a matching SOURCE address in the map entry. For outgoing calls, the PAD checks for a matching DESTINATION address in the map entry.

The **x25 map pad** commands are used to configure PAD and protocol translation accesses. They are also used to override the configuration of the interface on a per-destination basis.

The following example shows how to configure an X.25 interface to restrict incoming PAD access to a single mapped host. This example requires that both incoming and outgoing PAD accesses use the Network User Identification (NUID) to authenticate the user.

```
interface serial 0
 x25 pad-access
 x25 smap pad 219104 nuid johndoe secret
```

Accepting a PAD Call

An incoming PAD call is accepted by a Cisco router if the destination address matches the following criteria:

- A translation entry.
- The interface address.
- An alias of an interface.
- The address of the interface with trailing zeros.
- An interface subaddress.
- A NULL address.
- The address for the router set by the **x25 host** command.

When a Cisco router receives a call that requires protocol translation, the protocol translator searches the translation table for an entry with a regular expression in the X.121 address and the CUD field that matches the incoming X.121 address and the user data part of the CUD (the default PAD PID is not included).

If the PID is a nonstandard value (not equal to 0x01000000), the protocol translator searches the translation table for an entry with a regular expression in the X.121 address and the CUD field that matches the entire CUD (PID and user data).

For example, an incoming call to destination 417262510195 with a standard PAD PID of 0x01000000 and no user data will match the following translation entry:

```
translate x25 417262510195 tcp 172.31.186.54
```

An incoming call to destination 417262510195 with an unknown PID of 1234 and user data zayna will match the following translation entry:

```
translate x25 417262510195 cud 1234zayna tcp 172.31.186.54
```

An incoming call to destination 417262510195 with a standard PAD PID of 0x01000000 and user data zayna will match the following translation entry:

```
translate x25 417262510195 cud zayna tcp 172.31.186.54
```

Note Using the **translate** command, you can specify the CUD field in ASCII, octal, or hexadecimal format. You cannot enter CUD values in hexadecimal format using the **pad** command. However, you can enter the octal equivalents of CUD hexadecimal values using the following command syntax:

```
pad x121-address /cud \307\021
```

In the following example, the regular expression CUD field allows an incoming call to destination 31200100994301 with a standard PAD PID of 0x01000000 and User Data 0xD0<*whatever*> to match the following translation entry:

```
translate X25 31200100994301 cud \320.* tcp 172.20.169.11 port 13301
```

Note The PID cannot be eliminated. The entire CUD field cannot be 0. The PAD uses the PID length to determine if a PID was entered. Therefore, using the characters *""* or *\000* will be interpreted as if no PID was given.

Processing Outgoing PAD Calls Initiated by Protocol Translation

Specifying the use-map Option on Outgoing PAD and Protocol Translation Connections

Specifying the **use-map** option on the **pad** EXEC command or the **translate** global configuration command (as an outgoing protocol option) allows the optional PID, CUD, and facilities to be applied on a per-PAD connection or protocol-translation basis. If you specify the **use-map** option on the PAD connection or on the **translate** command, the DESTINATION address and (optional) PID and CUD are checked against a list of entries configured with the **x25 map pad** command.

When a match is found and the corresponding interface is available (up), the call is placed on that interface and the **x25 map** options, including facilities, are applied on the outgoing call. Otherwise, the PAD call is refused.

Note The **use-map** option is not supported on outgoing protocol translation PVCs.

For example, entering the **use-map** option on the **pad** EXEC command returns the following:

```
interface serial 1
 encapsulation x25
 x25 address 2192222
 x25 win 7
 x25 wout 7
 x25 ips 256
 x25 ops 256
 x25 map pad 77630 packetsize 1024 1024 windowsize 2 2 reverse
```

The interface in this example is configured for a window size of 7 and a packet size of 256.

The following example specifies the **use-map** option so that the outgoing PAD connection will override the interface facilities and apply a window size of 2, a packet size of 1024, and reverse charging on the outgoing PAD call:

```
pad 77630 /use-map
```

The following example specifies the **use-map** option so that a translation of the following outgoing PAD connection will cause the Call Request to be sent with a standard PAD PID and the user data to be sent in hexadecimal format:

```
! On the interface the call goes out on:
interface Serial1
 x25 map pad 417262510197 pid 0x01000000<hex for your user data>
!
translate tcp 172.21.186.54 x25 417262510197 use-map
```

The following example specifies the **use-map** options so that this outgoing PAD connection will cause the Call Request to be sent with a nonstandard PAD PID of 0x0E and user data hello:

```
! On the interface the call goes out on:
interface Serial1
 x25 map pad 417262510198 pid 0x0E cud hello
!
translate tcp 172.21.186.54 x25 417262510198 use-map
```

Applying the X.25 Route Table on Outgoing PAD and Protocol Translation Connections

When the **use-map** option is not specified on the **pad** EXEC command or the **translate** global configuration command as an outgoing protocol option, the PAD or the protocol translator locates the X.121 destination address in the X.25 route table to determine the interface on which to establish the outgoing switched virtual circuits (SVC) or permanent virtual circuits (PVCs). The destination address and optional CUD are checked against the configured list of X.25 route entries. If a matching route entry is found and the corresponding interface is operational, the call is placed on that interface. If the interface is not operational or out of available virtual circuits, the lookup for the next matching route is continued.

If the route disposition is clear, the PAD call is refused. If the route lookup does not match any valid entry, the call is placed on the first configured X.25 interface. If the default interface (that is, the first configured X.25 interface, which may or may not be up or available) is not operational or out of available virtual circuits, the PAD call is refused.

Increasing or Decreasing the Number of Virtual Terminal Lines

Because each protocol translation session uses a vty, you need to increase the number of virtual terminal lines to increase the number of protocol translation sessions. That is, if your router has ten virtual terminal lines, you can have ten protocol translation sessions. The default number of virtual terminal lines is 5 (lines 0 through 4). To increase the number of lines, and thus the maximum number of protocol translation sessions, use the following commands as needed, in global configuration mode:

Command	Purpose
`Router(config)# line vty line-number`	Increases the number of virtual terminal lines.
`Router(config-line)# no line vty line-number`	Decreases the number of virtual terminal lines.

Caution Protocol translation is a CPU-intensive task. Increasing the number of protocol translation sessions while routing is enabled can impact available memory. The amount of memory available depends on the platform type, the amount of DRAM available, the activity of each translation session, and the speed of the link. If you are using the maximum number of sessions and have problems with memory, you might need to decrease the number of protocol translation sessions.

The maximum number of protocol translation sessions for each platform can be increased to the number specified in Table 1. One virtual terminal is required for each protocol translation session.

Table 1 Maximum Number of Protocol Translation Sessions by Platform

Platform	Default Number of Virtual Terminal Lines	Total Number of Lines[1]	Maximum Virtual Terminal Lines with Translation Option
Cisco 1000 running Cisco IOS software	5	6	5
Cisco 2500 series (8 asynchronous ports)	5	200	180
Cisco 2500 series (16 asynchronous ports)	5	200	182
Cisco 2600 series	5	200	182
Cisco 3000 series	5	200	198
Cisco 3640	5	1002	872
Cisco 3620	5	1002	936
Cisco 4000 series	5	200	198
Cisco 4500 series	5	1002	1000
Cisco 4700 series	5	1002	1000
Cisco AS5200	5	200	182
Cisco AS5300	5	1002	952
Cisco 7000 series	5	120	118
Cisco 7200 series	5	1002	1000
Cisco 7000 series with RSP	5	1002	1000

1. Maximum number of virtual terminal lines = (TTYs + AUX + CON lines). Maximum number of virtual terminal lines with protocol translation option = (TTYs + AUX + CON lines).

Maintaining Virtual Interfaces

Monitoring and Maintaining a Virtual Access Interface

When a virtual interface template is applied to a protocol translation session, a virtual access interface is created dynamically. This is the only way a virtual access interface can be created. To display or clear a specific virtual access interface, use any of the following commands in user EXEC mode:

Command	Purpose
Router> **show users** [**all**]	Identifies the number associated with the virtual access interface, so you can display statistics about the interface or clear the interface.
Router> **show interfaces virtual-access** *number*	Displays the configuration of the virtual access interface.
Router> **clear interface virtual-access** *number*	Tears down the virtual access interface and frees the memory for other dial-in uses.

Displaying a Virtual Asynchronous Interface

To view information about the vty when the configuration of a virtual interface template is cloned to a vty configured as a virtual access interface for two-step protocol translation, use the following command in EXEC mode:

Command	Purpose
Router> **show line** [*line-number*]	Displays statistics about a vty.

Troubleshooting Virtual Asynchronous Interfaces

The following example shows the **debug** command output for the router redmount. It also shows the output for a specific **vty-async** interface. The **vty-async** command configures all virtual terminal lines on a router to support asynchronous protocol features.

```
Router# show debug

  PPP:
    PPP protocol negotiation debugging is on
  Asynchronous interfaces:
    Async interface framing debugging is on
    Async interface state changes debugging is on
  ROUTER1#
  ROUTER1#
  Initializing ATCP
  VTY-Async3: Set up PPP encapsulation on TTY3
  VTY-Async3: Setup PPP framing on TTY3
  VTY-Async3: Async protocol mode started for 172.22.164.1
  %LINK-3-UPDOWN: Interface VTY-Async3, changed state to up
  ppp: sending CONFREQ, type = 2 (CI_ASYNCMAP), value = A0000
  ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 91B8C7
  ppp: sending CONFREQ, type = 2 (CI_ASYNCMAP), value = A0000
  ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 91B8C7
  ROUTER1# debug 0x2
  ppp: config ACK received, type = 2 (CI_ASYNCMAP), value = A0000
  ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 91B8C7
  ppp: config ACK received, type = 7 (CI_PCOMPRESSION)
```

```
  ppp: config ACK received, type = 8 (CI_ACCOMPRESSION)
  PPP VTY-Async3: received config for type = 0x1 (MRU) value = 0x5DC acked
  PPP VTY-Async3: received config for type = 0x2 (ASYNCMAP) value = 0x0 acked
  PPP VTY-Async3: received config for type = 0x7 (PCOMPRESSION) acked
  PPP VTY-Async3: received config for type = 0x8 (ACCOMPRESSION) acked
  ipcp: sending CONFREQ, type = 3 (CI_ADDRESS), Address = 190.0.2.255
  ppp VTY-Async3: ipcp_reqci: rcvd COMPRESSTYPE (rejected) (REJ)
  ppp VTY-Async3: Negotiate IP address: her address 10.1.1.1 (NAK with address
172.22.164.1) (NAK)
  ppp: ipcp_reqci: returning CONFREJ.
  PPP VTY-Async3: state = REQSENT fsm_rconfack(0x8021): rcvd id 0x1
  ipcp: config ACK received, type = 3 (CI_ADDRESS), Address = 172.21.213.7
  ppp VTY-Async3: Negotiate IP address: her address 10.1.1.1 (NAK with address
172.22.164.1) (NAK)
  ppp: ipcp_reqci: returning CONFNAK.
  ppp VTY-Async3: Negotiate IP address: her address 172.22.164.1 (ACK)
  ppp: ipcp_reqci: returning CONFACK.
  %LINEPROTO-5-UPDOWN: Line protocol on Interface VTY-Async3, changed state to up

  Router# show interface vty-async 3

  VTY-Async3 is up, line protocol is up
    Hardware is Virtual Async Serial
    Interface is unnumbered.  Using address of Ethernet0 (172.21.213.7)
    MTU 1500 bytes, BW 9 Kbit, DLY 100000 usec, rely 255/255, load 1/255
    Encapsulation PPP, loopback not set, keepalive set (10 sec)
    DTR is pulsed for 0 seconds on reset
    lcp state = OPEN
    ncp ccp state = NOT NEGOTIATED   ncp ipcp state = OPEN
    ncp osicp state = NOT NEGOTIATED   ncp ipxcp state = NOT NEGOTIATED
    ncp xnscp state = NOT NEGOTIATED   ncp vinescp state = NOT NEGOTIATED
    ncp deccp state = NOT NEGOTIATED   ncp bridgecp state = NOT NEGOTIATED
    ncp atalkcp state = NOT NEGOTIATED   ncp lex state = NOT NEGOTIATED
    ncp cdp state = NOT NEGOTIATED
    Last input 0:00:01, output 0:00:02, output hang never
    Last clearing of "show interface" counters never
    Input queue: 1/75/0 (size/max/drops); Total output drops: 0
    Output queue: 0/64/0 (size/threshold/drops)
       Conversations  0/1 (active/max active)
       Reserved Conversations 0/0 (allocated/max allocated)
    5 minute input rate 0 bits/sec, 0 packets/sec
    5 minute output rate 0 bits/sec, 0 packets/sec
       26 packets input, 1122 bytes, 0 no buffer
       Received 0 broadcasts, 0 runts, 0 giants
       0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
```

Monitoring Protocol Translation Connections

This section describes how to log significant virtual terminal-asynchronous authentication information such as the X.121 calling address, CUD, and IP address assigned to a virtual terminal asynchronous connection. Depending on how you configure the logging information to be displayed, you can direct this authentication information to the console, an internal buffer, or a UNIX syslog server. This authentication information can be used to associate an incoming PAD virtual terminal-asynchronous connection with an IP address.

Note By default, Cisco IOS software displays all messages to the console terminal.

To monitor protocol translation connections, perform the following tasks:

Logging vty-Asynchronous Authentication Information to the Console Terminal

To log significant vty-asynchronous authentication information to the console terminal, use the following command in global configuration mode:

Command	Purpose
`Router(config)# service pt-vty-logging`	Logs significant virtual terminal-asynchronous authentication information.

Logging vty-Asynchronous Authentication Information to a Buffer

To log significant vty-asynchronous authentication information to a buffer, use the following commands in global configuration mode as needed:

	Command	Purpose
Step 1	`Router(config)# service pt-vty-logging`	Logs significant virtual terminal-asynchronous authentication information.
Step 2	`Router(config)# logging buffered [size]`	Directs the authentication log information to a buffer.

Logging vty-Asynchronous Authentication Information to a UNIX Syslog Server

To log significant vty-asynchronous authentication information to a UNIX syslog server, use the following commands in global configuration mode as needed:

	Command	Purpose
Step 1	`Router(config)# service pt-vty-logging`	Logs significant vty-asynchronous authentication information.
Step 2	`Router(config)# logging host`	Directs the authentication log information to a UNIX syslog server.

Troubleshooting Protocol Translation

To troubleshoot your protocol translation sessions, use the following **show** and **debug** commands:

- **debug async**
- **debug pad**
- **show arap**
- **show async status**

- **show interfaces virtual-access**
- **show ip local pool**
- **show line**

Use these commands in EXEC mode. Refer to Cisco IOS command references for explanations of command output.

Virtual Template for Protocol Translation Examples

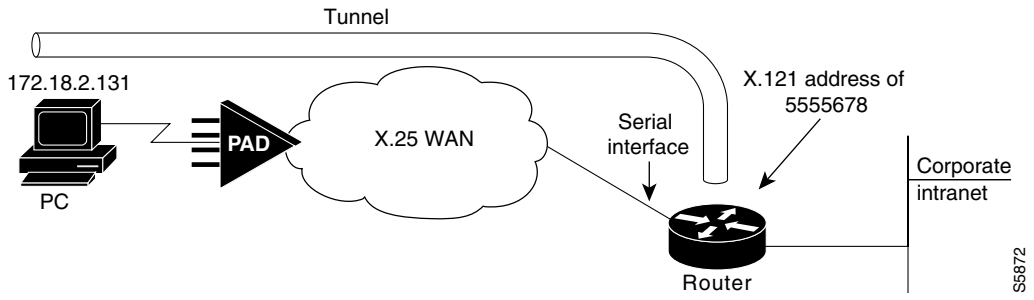
One-Step Examples

Tunnel PPP Across X.25: Example

The following example shows a virtual interface template that specifies a peer IP address of 172.18.2.131, which is the IP address of the PC in Figure 3. The virtual interface template explicitly specifies PPP encapsulation. The translation is from X.25 to PPP, which enables tunneling of PPP across an X.25 network, as shown in Figure 3.

```
interface virtual-template 1
 ip unnumbered Ethernet0
 ! Static address of 172.18.2.131 for the PC dialing in to the corporate intranet.
 peer default ip address pool group1
 ! Where the pool name is defined as ip local pool group1 172.18.35.1 172.18.35.5.
 encapsulation ppp
! X.121 address of 5555678 is the number the PAD dials to connect through the router.
translate x25 5555678 virtual-template 1
```

Figure 3 Tunneling PPP Across an X.25 Network



Tunnel SLIP Across X.25: Example

The following example uses SLIP encapsulation instead of PPP encapsulation on the virtual interface:

```
interface Virtual-Template5
 ip unnumbered Ethernet0
 encapsulation slip
 peer default ip address pool group1
 ! Where the pool name is defined as ip local pool group1 172.18.35.11 172.18.35.15.
!
translate x25 5555000 virtual-template 5
```

Tunnel PPP Across X.25 and Specify CHAP and Access List Security: Example

The following example uses PPP encapsulation on the virtual terminal interface, although it is not explicitly specified. It also uses CHAP authentication and an X.29 access list.

```
x29 access-list 1 permit ^5555
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 peer default ip address pool group1
 ! Where the pool name is defined as ip local pool group1 172.18.35.21 172.18.35.25.
 ppp authentication chap
!
translate x25 5555667 virtual-template 1 access-class 1
```

Tunnel PPP with Header Compression On: Example

The following example uses TCP header compression when tunneling PPP across X.25:

```
interface Virtual-Template1
 ip unnumbered Ethernet0
 ip tcp header-compression passive
 peer default ip address pool group1
 ! Where the pool name is defined as ip local pool group1 172.18.35.31 172.18.35.35.
!
translate x25 5555676 virtual-template 1
```

Tunnel IPX-PPP Across X.25: Example

The following example shows how to tunnel IPX-PPP across the X.25 network. It creates an internal IPX network number on a loopback interface, and then assigns that loopback interface to the virtual interface template.

```
ipx routing 0000.0c07.b509
!
interface loopback0
 ipx network 544
 ipx sap-interval 2000
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 ipx ppp-client Loopback0
 peer default ip address pool group1
 ! Where the pool name is defined as ip local pool group1 172.18.35.41 172.18.35.45.
!
translate x25 5555766 virtual-template 1
```

Two-Step Examples

Two-Step Tunneling of PPP with Dynamic Routing and Header Compression: Example

The following example uses the default PPP encapsulation on the virtual template.

```
vty-async
vty-async virtual-template 1
vty-async dynamic-routing
vty-async header-compression
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 no peer default ip address
```

After users connect to the router (in this example, named waffler), they invoke the **ppp** command to complete the two-step connection:

```
Router> ppp /routing /compressed 172.16.2.31
Entering PPP routing mode.
Async interface address is unnumbered (Ethernet0)
Your IP address is 172.16.2.31. MTU is 1500 bytes
```

Two-Step Tunneling of PPP with Dynamic Routing, TACACS, and CHAP: Example

The virtual template interface in the following example uses the default encapsulation of PPP and applies CHAP authentication with TACACS+:

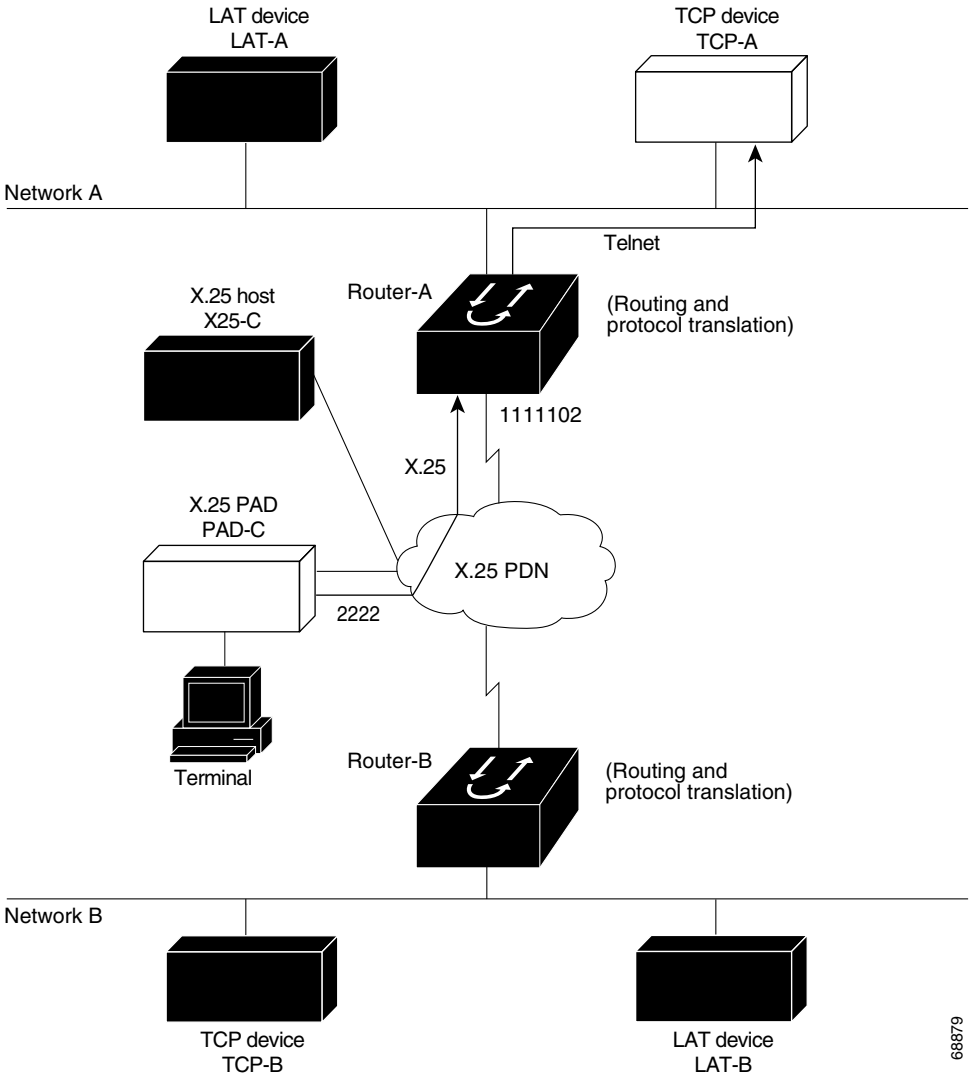
```
aaa authentication ppp default tacacs+
!
vty-async
vty-async dynamic-routing
vty-async virtual-template 1
!
interface Ethernet0
 ip address 10.11.12.2 255.255.255.0
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 no peer default ip address
 ppp authentication chap
```

Protocol Translation Application Examples

X.25 PAD-to-TCP Configuration: Example

Making a translated connection from an X.25 PAD to a TCP device is analogous to the preceding X.25 PAD-to-LAT example. (See Figure 4.) Instead of translating to LAT, the configuration for Router-A includes a statement to translate to TCP (Telnet). Note that a router with the protocol translation software option can include statements supporting both translations (X.25 PAD to LAT and X.25 PAD to TCP). Different users on the same PAD can communicate with X.25, LAT, or TCP devices.

Figure 4 X.25 PAD-to-TCP Translation



The following example shows how to use the **translate** global configuration command to translate from an X.25 PAD to a TCP device on Network A. It is applied to Router-A.

```
! Set up translation.
translate x25 2222 tcp TCP-A
```

Protocol Translation Session Examples

One-Step Method for TCP-to-X.25 Host Connections: Example

This example demonstrates one-step protocol translation featuring a UNIX workstation user making a connection to a remote X.25 host named host1 over an X.25 PDN. The router automatically converts the Telnet connection request to an X.25 connection request and sends the request as specified in the system configuration.

A connection is established when you enter the **telnet** EXEC 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.

The router accepts the Telnet connection and immediately forms an outgoing connection with remote host1 as defined in a **translate** command.

Next, host1 sets several X.3 parameters, including local echo. Because 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 username, then host1 sets the X.3 parameters to cause remote echo (the same process as setting X.3 PAD parameter 2:0), and prompts for a password. Cisco IOS software converts this request 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 finished with the connection, the user escapes back to the host connection, and then enters the appropriate command to close the connection.

The host named host1 immediately closes the X.25 connection. The Cisco IOS software then drops the TCP connection, leaving the user back at the UNIX system prompt.

Using the Two-Step Method for TCP-to-PAD Connections: Example

To use the two-step method for making connections, perform the following steps:

Step 1 Connect directly from a terminal or workstation to a router.

For example, you might make the following connection requests at a UNIX workstation as a first step to logging in to the database named Information Place on an X.25 PDN:

```
unix% telnet orion
```

If the router named orion is accessible, it returns a login message, and you can enter your login name and password.

Step 2 Connect from the router to Information Place, which is on an X.25 host. You connect to an X.25 host using the **pad** EXEC command followed by the service address:

```
Router> pad 71330
```

Once the connection is established, the router immediately sets the PAD to single-character mode with local echoing, because these are the settings the router expects. The PAD responds with its login messages and a prompt for a password:

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

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

To complete this sample session, log out to return to the router system EXEC prompt and enter the EXEC **quit** command; the router drops the network connection to the PAD.

Two-Step Protocol Translation for TCP-to-PAD Connections: Example

The following example shows a connection from a local UNIX host named host1 to a router named router1 as the first step in a two-step translation process:

```
host1% telnet Router1
```

The following sample session shows a connection from Router1 to a host named ibm3278 as the second step in a two-step translation process:

```
Router1> tn3270 ibm3278
ibm3278%
```

Next, connect directly from a terminal or workstation on a TCP/IP network to a router, and then to a database named Information Place on an X.25 packet data network. The database has a service address of 71330.

To complete the two-step translation connection, perform the following steps:

Step 1 Make the following connection requests at a UNIX workstation as a first step to logging in to the database Information Place:

```
unix% telnet router1
```

If the router named router1 is accessible, it returns a login message and you can enter your login name and password.

Step 2 Connect from the router to the Information Place, which is on an X.25 host. You connect to an X.25 host using the **pad** EXEC command followed by the service address:

```
Router1> pad 71330
```

Once the connection is established, the router immediately sets the PAD to single-character mode with local echoing, because these are the settings that the router expects. The PAD responds with its login messages and a prompt for a password.

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

Because the password should not echo on your terminal, the PAD requests remote echoing so that characters will be exchanged between the PAD and the router, but not echoed locally or displayed. After the password is verified, the PAD again requests local echoing from the router.

Step 3 Complete the session by logging out, which returns you to the router system EXEC prompt.

Step 4 Enter the **quit** EXEC command; the router drops the network connection to the PAD.

Changing Parameters and Settings Dynamically: Example

The following sample session shows how to make a dynamic change during a protocol translation session. In this sample, you will edit information on the remote host named Information Place. To change the X.3 PAD parameters that define the editing characters from the default Delete key setting to the Ctrl-D sequence, perform the following steps:

Step 1 Enter the escape sequence to return to the system EXEC prompt:

```
Ctrl ^ x
```

Step 2 Enter the **resume** command with the **/set** keyword and the desired X.3 parameters. X.3 parameter 16 sets the Delete function. ASCII character 4 is the Ctrl-D sequence.

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

The session resumes with the new settings, but the information is not displayed correctly. You may want to set the **/debug** switch to check that your parameter setting has not been changed by the host PAD.

Step 3 Enter the escape sequence to return to the system EXEC prompt, and then enter the **resume** command with the **/debug** switch.

```
Router> resume /debug
```

The **/debug** switch provides helpful information about the connection.

You can also set a packet dispatch character or sequence using the **terminal dispatch-character** command. The following example shows how to set ESC (ASCII character 27) as a dispatch character:

```
Router> terminal dispatch-character 27
```

To return to the PAD connection, enter the **resume** command:

```
Router> resume
```

Monitoring Protocol Translation Connections: Example

The following example shows how to log significant virtual terminal-asynchronous authentication information, such as the X.121 calling address, CUD, and IP address assigned to a virtual terminal-asynchronous connection, to a UNIX syslog server named alice:

```
service pt-vty-logging
logging alice
```

Two-Step Protocol Translation for Virtual Terminal Asynchronous Interfaces: Example

Caution The following example shows how to configure the **vty-async** command for PPP over X.25 using the router named redmount:

```
hostname redmount

ip address-pool local
x25 routing
vty-async                   <------ two-step translation
vty-async dynamic-routing   <------ optional
vty-async mtu 245           <------ optional

interface Ethernet0
 ip address 172.31.113.7 255.255.255.0
 no mop enabled

interface Serial0
 no ip address
 encapsulation x25
 x25 address 9876543210

router rip
 network 172.31.213.0
 network 172.22.164.0

ip domain-name cisco.com
ip name-server 172.31.213.2
ip name-server 172.31.213.4
ip local pool default 172.22.164.1 172.28.164.254
x25 route 9876543211 alias serial 0
x25 route 9876543212 alias serial 0

line con 0
 exec-timeout 0 0
line aux 0
 transport input all
line vty 0 1                <------ used for remote access to the router
 rotary 2
line vty 2 64               <------ used for ppp over x25
 rotary 1
 autocommand ppp default
```

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© 2011 Cisco Systems, Inc. All rights reserved.

Authorization for Protocol Translation

In releases of Cisco IOS software prior to 12.3(2)T, protocol translation sessions established using one-step protocol translation are set up without an authorization request being issued first. The Authorization for Protocol Translation feature adds an option to require that an authorization request is issued as a prerequisite to establishing a protocol translation session. This feature improves authentication, authorization, and accounting (AAA) support for protocol translation.

Feature History for the Authorization for Protocol Translation Feature

Release	Modification
Cisco IOS XE Gibraltar 16.10.1	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

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

Prerequisites for Authorization for Protocol Translation

Packet assembler/disassembler (PAD) must be configured. For more information on configuring PAD, refer to Configuring the Cisco PAD Facility for X.25 Connections.

A TACACS+ server must be configured to perform authorization. For more information about configuring authorization, refer to the "Configuring Authorization" chapter in the *Cisco IOS Security Configuration Guide*.

Restrictions for Authorization for Protocol Translation

This feature is supported only for protocol translation sessions in which the incoming protocol is TCP or X.25, and in which the outgoing protocol is TCP, X.25, or autocommand.

For incoming X.25 sessions, this feature is restricted to switched virtual circuits (SVCs) only; permanent virtual circuits (PVCs) may be used only for the outgoing side.

If the **pvc** keyword is specified in the **translate** command, the **authorize** and **login** keywords may not be used.

Information About Authorization for Protocol Translation

To configure the Authorization for Protocol Translation feature, you must understand the following concepts:

- AAA Authorization and the Authorization Packet, page 2
- Benefits of Authorization for Protocol Translation, page 2

AAA Authorization and the Authorization Packet

Once authorization is enabled, authorization occurs before access to the connection is granted. If authentication is configured, authorization occurs after authentication.

During authorization, a TACACS+ authorization packet is generated. This authorization packet contains the following attribute-value (AV) pairs:

- service—A new value, translate, has been added to the existing service AV pair defined in the args section. This AV pair is marked as mandatory.
- azn-tag—This new attribute contains the authorization tag assigned to the command. The azn-tag attribute may contain a series of lowercase alphanumeric ASCII characters up to 64 bytes in length. Allowable characters are digits, lowercase letters, the hyphen, and the underscore. This AV pair is marked as mandatory.

Benefits of Authorization for Protocol Translation

Releases of Cisco IOS software prior to 12.3(2)T did not allow authorization of protocol translation sessions established using one-step protocol translation. The Authorization for Protocol Translation feature introduces the ability to configure one-step protocol translation sessions for AAA authorization using TACACS+. This feature improves AAA support for protocol translation sessions.

How to Configure Authorization for Protocol Translation

This section contains the following procedures:

Configuring Authorization for Protocol Translation for a TCP-to-X.25 Protocol Translation Session

Perform this task to enable AAA authorization of a TCP-to-X.25 protocol translation session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authorization network** {**default** | *list-name*} *method1* [*method2*...]
4. **translate tcp** *incoming-address* [*incoming-options*] **x25** *outgoing-address* [*outgoing-options*] [*global-options*] **authorize** *method-list tag*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**aaa authorization network** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Router(config)# aaa authorization network mylist group tacacs+`	Sets parameters that restrict user access to a network.
Step 4	**translate tcp** *incoming-address* [*incoming-options*] **x25** *outgoing-address* [*outgoing-options*] [*global-options*] **authorize** *method-list tag* **Example:** `Router(config)# translate tcp 10.60.155.63 pvc 3 dynamic x25 12345678 authorize mylist 05149c3`	Translates a connection request to another protocol connection type when receiving a TCP connection request to a particular destination address or host name. • **authorize**—Enables AAA authorization for a protocol translation session. • *method-list*—The list of authorization methods defined with the **aaa authorization** command using the **network** keyword. The *method-list* argument may have the value of *list-name* or **default**. • *tag*—An alphanumeric string of up to 64 characters. The *tag* argument need not be unique; more than one instance of the **translate** command can specify identical values for the *tag* argument. **Note** The *tag* argument is not interpreted by the router. The *tag* argument simply identifies the translate command or the resource being accessed by the protocol translation session to the authorization server.

Configuring Authorization for Protocol Translation for an X.25-to-TCP Protocol Translation Session

Perform this task to enable AAA Authorization of an X.25-to-TCP protocol translation session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authorization network** {**default** | *list-name*} *method1* [*method2...*]
4. **translate x25** *incoming-address* [*incoming-options*] **tcp** *outgoing-address* [*outgoing-options*] [*global-options*] **authorize** *method-list tag*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization network** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Router(config)# aaa authorization network mylist group tacacs+`	Sets parameters that restrict user access to a network.
Step 4	**translate x25** *incoming-address* [*incoming-options*] **tcp** *outgoing-address* [*outgoing-options*] [*global-options*] **authorize** *method-list tag* **Example:** `Router(config)# translate x25 12345678 tcp 10.60.155.63 login authorize mylist 73234r45`	Translates a connection request to another protocol connection type when receiving an X.25 connection request to a particular destination address or host name. • **authorize**—Enables AAA authorization for a protocol translation session. • *method-list*—The list of authorization methods defined with the **aaa authorization** command using the **network** keyword. The *method-list* argument may have the value of *list-name* or **default**. • *tag*—An alphanumeric string of up to 64 characters. The *tag* argument need not be unique; more than one instance of the **translate** command can specify identical values for the *tag* argument.

Configuration Examples for Authorization for Protocol Translation

This section contains the following configuration example:

- Configuring Translation Authorization for a TCP-to-X.25 Protocol Translation Session: Example, page 6
- Configuring Translation Authorization for an X.25-to-TCP Protocol Translation Session: Example, page 7

Configuring Translation Authorization for a TCP-to-X.25 Protocol Translation Session: Example

The following example uses an authorization method list named mygroup. Serial interfaces 2/0 and 2/1 connect to X.25 hosts, each of which provides multiple services at different X.25 subaddresses. Some of the translate statements specify unique authorization tags so the services can be individually controlled; others specify generic tags (perhaps because they are less critical, such as a monitoring service rather than one which permits configuration changes).

```
aaa authorization network mygroup group tacacs+
x25 routing
!
interface Ethernet0/0
 ip address 10.60.155.30 255.255.255.0
!
interface Serial2/0
 encapsulation x25 dce
 x25 ltc 30
!
interface Serial2/1
 encapsulation x25 dce
 x25 ltc 30
!
x25 route ^13033 interface Serial2/0
x25 route ^13133 interface Serial2/1
!
translate tcp 10.60.155.36 port 2001 x25 1303301 login authorize mygroup a-port01
translate tcp 10.60.155.36 port 2002 x25 1303302 login authorize mygroup a-port02
translate tcp 10.60.155.36 port 2003 x25 1303303 login authorize mygroup monitor
translate tcp 10.60.155.36 port 2004 x25 1303304 login authorize mygroup monitor
translate tcp 10.60.155.36 port 2005 x25 13033 pvc 1 login authorize mygroup a-admin01
!
translate tcp 10.60.155.36 port 2101 x25 1313301 login authorize mygroup b-port01
translate tcp 10.60.155.36 port 2102 x25 1313302 login authorize mygroup b-port02
translate tcp 10.60.155.36 port 2103 x25 1313303 login authorize mygroup monitor
translate tcp 10.60.155.36 port 2104 x25 1313304 login authorize mygroup monitor
```

With this configuration, the router accepts Telnet requests to 10.60.155.36 at any of the TCP ports listed. The user is required to log in, then the router sends an authorization request specifying “translate” as the value of the “service” AV pair, and the authorization tag from the corresponding **translate** command as the value of the “azn-tag” AV pair. The user id and remote address of the Telnet session are also included in the authorization request. If the authorization server approves the request, the connection to the specified X.25 address is attempted; if the request is denied, the Telnet connection is closed.

The authorization server would not be able to distinguish between connections to 10.60.155.36 port 2003 and 10.60.155.36 port 2104, because they specify the same authorization tag.

Configuring Translation Authorization for an X.25-to-TCP Protocol Translation Session: Example

The following example uses the default authorization method list. Incoming PAD calls to the router on serial interface 1/1 are translated to Telnet calls to various destinations based on the X.25 subaddress. Use of the first two translate statements is restricted to users that are approved by the authorization server for access to group1; the third translate statement will complete the connection only if the authorization server grants access to group2.

```
aaa authorization network default group tacacs+
!
interface Serial1/1
 encapsulation x25
 x25 address 5551088
!
translate x25 555108801 tcp 10.60.155.1 login authorize default group1
translate x25 555108802 tcp 10.60.155.2 login authorize default group1
translate x25 555108803 tcp 10.60.155.3 login authorize default group2
```

Additional References

The following sections provide additional information related to the Authorization for Protocol Translation feature.

Related Documents

Related Topic	Document Title
Information on configuring PAD	The "Configuring the Cisco PAD Facility for X.25 Connections" chapter in the *Cisco IOS Terminal Services Configuration Guide*
Additional PAD commands: complete command syntax, command mode, defaults, usage guidelines and examples	*Cisco IOS Terminal Services Command Reference*, Release 12.3
Additional information about configuring authorization	"Configuring Authorization" chapter in the *Cisco IOS Security Configuration Guide*
Additional authentication commands: complete command syntax, command mode, defaults, usage guidelines and examples	*Cisco IOS Security Command Reference*, Release 12.3

Standards

Standards	Title
None	—

MIBs

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

RFCs

RFCs	Title
None	—

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

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

- **aaa authorization**
- **translate tcp**
- **translate x25**

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2008 Cisco Systems, Inc. All rights reserved.

End-of-Record Function for DCNs

The Cisco Protocol Translator is designed to support Telnet-like applications that are stream-based, with no recognition or accommodation for logical records. For record-oriented applications, problems can occur because the record boundaries in X.25 data are lost when translation to TCP occurs. The End-of-Record Function for Data Communications Networks (DCNs) feature provides for the configuration of an End of Record (EOR) marker which allows the X.25 logical boundaries to be marked when translated to TCP. The benefit of this feature is that it allows the preservation of logical boundaries when translating X.25 data to TCP, enabling X.25-based networking solutions to adapt to and benefit from TCP/IP technologies.

Feature History for the End-of-Record Function for DCNs Feature

Release	Modification
Cisco IOS XE Gibraltar 16.10.1	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

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

Prerequisites for End-of-Record Function for DCNs

X.25 must be configured. For more information on configuring X.25, see the "Configuring X.25 and LAPB" chapter in the *Cisco IOS Wide-Area Networking Configuration Guide*.

Restrictions for End-of-Record Function for DCNs

This feature is supported only for X25-to-TCP and TCP-to-X.25 protocol translation sessions.

This feature is not supported for any other types of protocol translation sessions.

Information About End-of-Record Function for DCNs

To configure the End-of-Record Function for DCNs feature, you must understand the following concepts:

Data Types

X.25 data

X.25 data is inherently record-oriented. The X.25 protocol defines a bit in the packet called the More-bit (M-bit), which indicates whether the packet should be considered to terminate a logical record.

TCP data

TCP data is inherently stream-oriented. The TCP protocol attaches no significance to TCP segment stream boundaries, and the boundaries may change if the data is re-sent.

The EOR Marker

Logical record boundaries indicated by the combination of the X.25 packet boundaries and the M-bit are not preserved when translation to TCP occurs. The End-of-Record Function for DCNs feature allows the X.25 logical record boundaries to be marked by inserting a configurable string into the TCP stream at each X.25 record boundary. Translation of X.25 packets without the M-bit will invoke the insertion of the EOR marker.

Benefits of End-of-Record Function for DCNs

The benefit of the End-of-Record Function for DCNs feature is that it allows the preservation of logical boundaries when translating X.25 data to TCP, enabling X.25-based networking solutions to adapt to and benefit from TCP/IP technologies.

How to Configure End-of-Record Function for DCNs

This section contains the following procedures:

Configuring the End-of-Record Function for a TCP-to-X.25 Protocol Translation Session

Perform this task to enable the End-of-Record Function for DCNs feature for a TCP-to-X.25 protocol translation session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **translate tcp** *incoming-address* [*incoming-options*] **x25** *outgoing-address* [*outgoing-options*] [*global-options*] **eor** *marker* [**insert**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**translate tcp** *incoming-address* [*incoming-options*] **x25** *outgoing-address* [*outgoing-options*] [global-options] **eor** *marker* [**insert**] **Example:** `Router(config)# translate tcp 10.60.155.63 x25 12345678 pvc 3 dynamic eor 0x19 insert`	Translates an incoming TCP connection request to an X.25 destination address or host name and enables EOR functionality. • **eor** *marker*—Defines the EOR marker for the translation session. The *marker* argument may be any set characters from 1 to 4 in length. Nonprintable characters must be entered in hexadecimal format. Printable characters may be typed in. • **insert**—Allows the EOR marker to be inserted into the TCP stream after each received X.25 packet that does not contain the M-bit set.

Configuring the End-of-Record Function for an X.25-to-TCP Protocol Translation Session

Perform this task to enable the End-of-Record Function for DCNs feature for an X.25-to-TCP protocol translation session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **translate x25** *incoming-address* [*incoming-options* [**pvc** *number* [*pvc-options*]]] **tcp** *outgoing-address* [*outgoing-options*] [*global-options*] **eor** *marker* [**insert**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**translate x25** *incoming-address* [*incoming-options* [**pvc** *number* [*pvc-options*]]] **tcp** *outgoing-address* [*outgoing-options*] [global-options] **eor** *marker* [**insert**] **Example:** `Router(config)# translate x25 12345678 pvc 3 tcp 10.60.155.63 eor AAA insert`	Translates an incoming X.25 connection request to a TCP destination address or host name and enables EOR functionality. • **eor** *marker*—Defines the EOR marker for the translation session. The *marker* argument may be any set of characters from 1 to 4 in length. Nonprintable characters must be entered in hexadecimal format. Printable characters may be typed in. • **insert**—Allows the EOR marker to be inserted into the TCP stream after each received X.25 packet that does not contain the M-bit set.

Troubleshooting Tips

In the event that the End-of-Record Function for DCNs feature is not operating correctly, use the following **debug** commands in privileged EXEC mode to determine the source of the problem:

- **debug translate**
- **debug x25 all**
- **debug pad**

Refer to the *Cisco IOS Debug Command Reference* publication for information about the **debug translate**, **debug x25 all**, and **debug pad** commands.

Monitoring and Maintaining the End-of-Record Function for DCNs

This task results in the display of information about any protocol translation information configured with the **translate** command.

SUMMARY STEPS

1. **enable**
2. **show translate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show translate** **Example:** `Router# show translate`	Displays information about translation sessions that have been configured.

Configuration Examples for End-of-Record Function for DCNs

This section contains the following configuration example:

- Configuring the End-of-Record Function for DCNs for a TCP-to-X.25 Protocol Translation Session Example, page 5
- Configuring the End-of-Record Function for DCNs for an X.25-to-TCP Protocol Translation Session Example, page 6

Configuring the End-of-Record Function for DCNs for a TCP-to-X.25 Protocol Translation Session Example

The following example configures a TCP-to-X.25 protocol translation session to insert an EOR marker in a TCP packet after each received X.25 packet that does not contain the M-bit set. The EOR marker in this example consists of nonprintable characters and is entered in hexadecimal format.

```
translate tcp 10.60.155.63 x25 12345678 pvc 3 dynamic eor 0x19 insert
```

Configuring the End-of-Record Function for DCNs for an X.25-to-TCP Protocol Translation Session Example

The following example configures an X.25-to-TCP protocol translation session to insert an EOR marker in a TCP packet after each received X.25 packet that does not contain the M-bit set. The EOR marker in this example consists of printable characters.

```
translate x25 12345678 pvc 3 tcp 10.60.155.63 eor AAA insert
```

Additional References

The following sections provide additional information related to the End-of-Record Function for DCNs feature.

Related Documents

Related Topic	Document Title
Additional information about configuring protocol translation	"*Configuring Protocol Translation and Virtual Asynchronous Devices*
Additional protocol translation commands: complete command syntax, command mode, defaults, usage guidelines and examples	*Cisco IOS Terminal Services Command Reference*
Information on configuring X.25	*Cisco IOS Wide-Area Networking Configuration Guide.*
Additional X.25 commands	*Cisco IOS Wide-Area Networking Command Reference*

Standards

Standards	Title
None	—

MIBs

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

RFCs

RFCs	Title
None	—

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

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

- **show translate**
- **translate tcp**
- **translate x25**

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2008 Cisco Systems, Inc. All rights reserved.

Protocol Translation Ruleset

The Protocol Translation Ruleset feature provides an effective method for creating Cisco IOS protocol translation configurations by defining a set of statements called a *ruleset*. The ruleset applies pattern matching and substitution technology to use incoming protocol elements, such as a destination address and port, to determine the outgoing protocol elements and translation options specified for originated connections. The ruleset also contains options to control the protocol translation sessions. The Protocol Translation Ruleset feature is especially useful for users that need to configure a large number of translate commands, because it makes it easy to create many individual translate configuration commands using a single ruleset-based command.

Feature History for the Protocol Translation Ruleset Feature

Release	Modification
Cisco IOS XE Gibraltar 16.10.1	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

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

Prerequisites for Using the Protocol Translation Ruleset

Understanding how to compose regular expressions for matching patterns in Cisco IOS software configurations and scripts is key to understanding the Protocol Translation Ruleset feature. Composing regular expressions is described in the *Regular Expressions* document.

A protocol translation ruleset does not look up the X.25 route table for a matching destination entry. An interface on which to set up the permanent virtual circuit (PVC) must be specified. Protocol translation requires a client to register for PVCs that are available for protocol translation use, whether or not a session is active for the channel. Protocol translation ruleset processing introduced by the Protocol Translation Ruleset feature allows connections only to a PVC that has been reserved for ruleset handling. You must use the **x25 pvc translate ruleset** command to reserve the PVCs.

In a Telnet-to-PAD protocol translation ruleset, an IP address must be specified with the **translate use telnet** command for the protocol translator to respond to Address Resolution Protocol (ARP) attempts for that address. The IP address that the protocol translation software listens for must be on a connected subnet; it cannot be used by another interface unless you also specify a TCP port number, and there cannot be another host that responds to ARPs for that address.

Restrictions for a Protocol Translation Ruleset

The ruleset introduced in the Protocol Translation Ruleset feature allows dynamic construction of the information needed to configure a protocol translation session. It was designed specifically to increase the flexibility of these sessions, especially in large networks with an address plan that can make effective use of pattern matching capability. However, this increased functionality may overload router memory and processing capabilities if it generates large numbers of concurrent sessions or a high aggregate volume of traffic. Memory and performance impact will vary depending upon the particulars of network design and traffic load.

Information About the Protocol Translation Ruleset

Before starting the tasks described in this document, you need to understand the following concepts:

- Cisco IOS Protocol Translation and Translation by Ruleset, page 3
- Cisco Regular Expression Pattern Matching, page 3
- Regular Expression Pattern Matching in a Protocol Translation Ruleset, page 4
- Error Handling in the Protocol Translation Ruleset, page 6

Cisco IOS Protocol Translation and Translation by Ruleset

The Cisco IOS software provides protocol translation capability that can be used in many types of networks and translate between incoming connection protocols such as TCP/IP, X.25 packet assembler/disassembler (PAD), and local-area transport (LAT), and a set of outgoing protocols that includes TCP/IP, X.25 PAD, LAT, PPP, and Serial Line Internet Protocol (SLIP). Each translation configuration is entered as a single command line, and users can choose from a lengthy list of options to define configurations for specific environments. For some users, however, it is more important to be able to quickly and efficiently define translation connections for a large number of addresses. The Protocol Translation Ruleset feature provides this capability by defining Cisco IOS protocol translation configurations in a ruleset. The ruleset is defined by using regular expression pattern matching and operations that match or ignore incoming connection requests. Substitute, set, and test string writing operations create the connection configurations based on an incoming address. This combination of pattern matching and string writing operations makes it possible to convert, for example, an IP port number to an X.121 address using just a few statements, rather then enter each configuration statement on a separate line.

The protocol translation capability introduced in the Protocol Translation Ruleset feature for Cisco IOS Release 12.3(8)T supports protocol translation from PAD to TCP and from TCP to PAD. Options are available for translations created in the ruleset to define a maximum number of sessions, require login, match an access list, and that suppress translation information messages on the session.

The Cisco IOS Release 12.3(8)T software will accept both the single-line translate commands (such as **translate pad** and **translate tcp**) and their option settings, and protocol translation statements defined in a ruleset, in the same configuration file. The ruleset configuration is applied after the incoming protocol translation connections are tested against the single line translate command configuration, so that you can make use of both the robust protocol translation capability currently available in the Cisco IOS software, and of a protocol translation ruleset that allows quick configuration of a large number of addresses.

The new ruleset environment will seem familiar to users that already know Cisco's single-line translate commands, in that many of the same keywords that are available for these commands are also used in the protocol translation ruleset. A new global configuration command, **translate ruleset**, specifies a name for the ruleset, defines the direction of translation, either from PAD to TCP or from TCP to PAD, and starts translate ruleset configuration mode. The translate ruleset configuration mode allows much flexibility in the number of statements accepted on each line. The mode also accepts multiple statements of the same type. The translate ruleset configuration mode provides **match** and **skip** commands to create statements that look at incoming connection requests to determine if they are valid, and **substitute**, **set**, and **test** commands for string writing operations that will help configure the translation session.

To assist you with writing statements that configure the connections and options needed for your network, the Protocol Translation Ruleset feature provides the **test translate** and **show translate ruleset** privileged EXEC commands. The **test translate** command is interactive and will step through the command statements to test their validity. The **show translate ruleset** command displays information about the connection rulesets to help you modify and maintain them.

Cisco Regular Expression Pattern Matching

Table 1 summarizes the basic Cisco IOS regular expression characters and their functions.

Table 1 Cisco Regular Expression Characters

Regular Expression Character	Function	Examples
.	Matches any single character.	0.0 matches 0x0 and 020 t..t matches strings such as test, text, and tart
\	Matches the character following the backslash. Also matches (escapes) special characters.	172\.1\.. matches 172.1.10.10 but not 172.12.0.0 \. allows a period to be matched as a period
[ ]	Matches the characters or a range of characters separated by a hyphen, within left and right square brackets.	[02468a-z] matches 0, 4, and w, but not 1, 9, or K
^	Matches the character or null string at the beginning of an input string.	^123 matches 1234, but not 01234
$	Matches the character or null string at the end of an input string.	123$ matches 0123, but not 1234
*	Matches zero or more sequences of the character preceding the asterisk. Also acts as a wildcard for matching any number of characters.	5* matches any occurrence of the number 5 including none 18\..* matches the characters 18. and any characters that follow 18.
+	Matches one or more sequences of the character preceding the plus sign.	8+ requires there to be at least one number 8 in the string to be matched
() []	Nest characters for matching.	(17)* matches any number of the two-character string 17 ([A-Za-z][0-9])+ matches one or more instances of letter-digit pairs: b8 and W4, as examples
|	Concatenates constructs. Matches one of the characters or character patterns on either side of the vertical bar.	A(B|C)D matches ABD and ACD, but not AD, ABCD, ABBD, or ACCD

The order for matching using the * or + character is longest construct first. Nested constructs are matched from the outside in. Concatenated constructs are matched beginning at the left side. If a regular expression can match two different parts of an input string, it will match the earliest part first.

Regular Expression Pattern Matching in a Protocol Translation Ruleset

Regular expressions for the Protocol Translation Ruleset feature have two uses: They match a text string against a defined pattern, and they can use information from a defined regular expression match operation to create a different string using substitution. These operations are performed by combining the characters described in Table 1 with commands from the translate ruleset configuration mode.

To understand regular expression pattern matching, begin by using Table 1 to interpret the following regular expression statement to match a string starting with the characters 172.18.:

^172\.18\..*

The following regular expression statement matches a five-digit number starting with 10 or 11:

^1[0-1]...$

Consider the following set of actions in a ruleset named B. This ruleset listens for incoming Telnet connections from a particular IP address and port number but ignores (skips) others, decides which PAD destination address the matched incoming connections should be connected to, then finally sets the PAD connection's X.25 VC idle timer from the first digit of the port number.

translate ruleset B from telnet to pad

match dest-addr ^10.2.2.(..)$ dest-port ^20..$

skip dest-addr ^10.2.2.11$

set pad dest-addr 4444

substitute telnet dest-port ^200(.)$ into pad idle \1

The caret sign anchors a match to the beginning of a string, in this example, 10.2.2 for the destination address and 20 for the destination port.

The parentheses are a powerful tool for the regular expression match operation because they identify groups of characters needed for a substitution. Combined with the substitute...into statement, the parentheses can dynamically create a broad range of string patterns and connection configurations.

In the example, the periods in the parentheses pair can be thought of as placeholders for the characters to be substituted. The dollar sign anchors the substitution match to the end of a string. The backslash preceding the number makes it a literal setting, so no substitution will be done to the idle timer setting.

The **test translate ruleset** command tests the script, and for the previous example would provide a report like the following:

Translate From: Telnet 10.2.2.10 Port 2000

To: PAD 4444

Ruleset B

0/1 users active

Consider the following, more complex expression:

^172\.18\.(10)\.(.*)$.

This expression matches any string beginning with 172.18. and identifies two groups, one that matches 10 and the other that matches a wildcard character.

Let us say that the regular expression ^172\.18\.(10)\.(.*)$ matched the characters 172.18.10.255 from an incoming connection. Once the match is made, the software places the character groups 10 and 255 into buffers and writes the matched groups using a substitution expression.

Regular expression substitution into the expression 0001172018\1\2 would generate the string 000117201810255.

The regular expression \0 would write the entire matched string, and substitution into the expression 0001\0 would generate the string 0001172.18.10.255.

Error Handling in the Protocol Translation Ruleset

Configuration errors are not detected when translation ruleset commands are entered. They are tested when the connection is attempted or with the **test (ruleset)** EXEC command. In the following example, the set statement unconditionally sets the PAD's profile name to a profile that does not exist in the configuration:

```
set pad profile Bldg-1-5ess
```

This command would be accepted at the command-line interpreter, and validated only upon a connection attempt or with the **test (ruleset)** command. When the error is detected, the following messages display:

```
*%PT-3-PARAMRESULTERR: PT ruleset test protocol pad parameter profile parse error:
Bldg-1-5ess.
-Process= "PAD InCall", ipl= 3, pid= 94
*PAD: ruleset translation not generated Cause: 9 Diag: 0
```

How to Configure a Protocol Translation Ruleset

This section contains the following tasks:

- Configuring a PVC for Protocol Translation Rulesets, page 6 (required)
- Creating Protocol Translation Rulesets, page 7 (required)
- Testing and Maintaining Protocol Translation Rulesets, page 10 (optional)

Configuring a PVC for Protocol Translation Rulesets

The protocol translation rulesets make connections only to a PVC that has been reserved for ruleset handling. Perform the following task to reserve the PVCs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot/port*
4. **x25 pvc** *number* **translate ruleset** *name*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot/port* **Example:** `Router(config)# interface serial 2/0`	Configures an interface type and enters interface configuration mode.
Step 4	**x25 pvc** *number* **translate ruleset** *name* **Example:** `Router(config-if)# x25 pvc 4 translate ruleset A`	Configures a PVC that is valid for protocol translation ruleset handling.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits the current configuration mode.

What to Do Next

Perform the tasks in the section "Creating Protocol Translation Rulesets" to create the protocol translation rulesets that configure protocol translation connections. Your rulesets may be simpler or more complex than those shown in the "Configuration Examples for the Protocol Translation Ruleset Feature" section on page 11, depending upon the requirements of your network.

Creating Protocol Translation Rulesets

This section describes how to create the protocol translation rulesets.

Components of a Ruleset

A protocol translation ruleset is defined by using a combination of pattern matching and commands that match or skip incoming connection requests, and then write connection configuration statements using substitute, test, and set operations. For example, telco customers that need many unique connections based on the telephone numbers in an exchange can use rulesets to generate the hundreds of specific commands as connections are established. Each generated command guides the interface and switched virtual circuit (SVC) or PVC assignment based on the incoming IP address and port selection elements.

You create the protocol translation rulesets in translate ruleset configuration mode, which is accessed when you issue the **translate ruleset** global configuration command. You define the ruleset name and the incoming and outgoing protocols to be translated using commands available in the translate ruleset configuration mode.

Numerous configuration options can be entered as part of the translation ruleset, and these options are described in the command pages for the **translate ruleset** global configuration command, and the **description**, **match**, **options**, **set**, **skip**, **substitute**, and **test** translate ruleset configuration commands.

Prerequisites

You must understand how to compose statements using Cisco regular expressions for matching patterns in a translation ruleset. See the instructions for composing regular expressions in the "Regular Expression Pattern Matching in a Protocol Translation Ruleset" section on page 4 in this document, or the "Related Documents" section on page 15.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **translate use telnet** *ip-address* (used only for Telnet-to-PAD translations statements)
4. **translate ruleset** *name* **from** *incoming-protocol* **to** *outgoing-protocol*
5. **description** *text*
6. {**match** | **skip**} [*line-number*] *incoming-connection-parameter regular-expression* [*line-number incoming-connection-parameter regular-expression* [...]]
7. **substitute** [*line-number*] {**pad** | **telnet**} *variable-parameter reg-exp-match* **into** {**pad** | **telnet**} *variable-parameter* [*reg-exp-write*]
8. **test** [*line-number*] {**pad** | **telnet**} *variable-parameter reg-exp-match* [{**pad** | **telnet**} *variable-parameter reg-exp-match* [...]] **set** {**pad** | **telnet**} *variable-parameter*
9. **set** [*line-number*] {**pad** | **telnet**} *variable-parameter* [{**pad** | **telnet**} *variable-parameter* [...]]
10. **options** *rule-option value* [*rule-option value* […]]
11. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**translate use telnet** *ip-address* **Example:** `Router(config)# translate use telnet 172.30.20.15`	Specifies a required IP address in a Telnet-to-PAD protocol translation ruleset. **Note** Required only for Telnet-to-PAD translation statements.
Step 4	**translate ruleset** *name* **from** *incoming-protocol* **to** *outgoing-protocol* **Example:** `Router(config)# translate ruleset Telnet-PAD from telnet to PAD`	Defines a unique name for a translation ruleset, specifies the translated protocols, and enters translate ruleset configuration mode. • **from** *incoming-protocol*—Choose **telnet** or **pad** • **to** *outgoing-protocol*—Choose **telnet** or **pad**
Step 5	**description** *text* **Example:** `Router(cfg-pt-ruleset)# description Template Telnet-PAD for site 101`	Adds a description about a translation ruleset.

Step 6	{**match** | **skip**} [*line-number*] *incoming-connection-parameter regular-expression* [*line-number incoming-connection-parameter regular-expression* [...]]
	
	Example:
	`Router(cfg-pt-ruleset)# skip source-addr ^10\.*`
	
	and
	
	`Router(cfg-pt-ruleset)# match dest-addr ^172\.30\..* dest-port ^12[0-7]..$`
		Identifies a connection for processing by a protocol translation ruleset.
		• Use regular expressions to write a **match** or **skip** statement that will look at incoming connection addresses.
		• Up to six **match** or **skip** statements can be entered on the command line, and multiple **match** statements can be entered in the ruleset.
		Note Each protocol translation ruleset must have at least one **match** statement.

Step 7	**substitute** [*line-number*] {**pad** | **telnet**} *variable-parameter reg-exp-match* **into** {**pad** | **telnet**} *variable-parameter* [*reg-exp-write*] **Example:** `Router(cfg-pt-ruleset)# substitute telnet dest-port ^.(...). into pad source-addr`	Matches an available protocol and substitutes another into the translation ruleset. • Use this command to substitute between protocol parameters using regular expressions to match elements with a test string, and to substitute parameters into another string that can take elements from the matched string. • A **substitute** … **into** statement will perform a regular expression match on any available protocol parameter and, if matched, substitute into any available protocol parameter. • Up to six **substitute** statements can be entered on the command line, and multiple **substitute** statements can be entered in the ruleset.

	Command or Action	Purpose
Step 8	**test** [*line-number*] {**pad** | **telnet**} *variable-parameter reg-exp-match* [{**pad** | **telnet**} *variable-parameter reg-exp-match* [...]] **set** {**pad** | **telnet**} *variable-parameter* **Example:** `Router(cfg-pt-ruleset)# test telnet dest-addr ^172\.30\.0\.* telnet dest-port ^10.00 \ set pad pvc 1 telnet binary T`	Tests parameter values in a translation ruleset using regular expressions. • A **test** … **set** statement conditionally sets one or more connection parameters to a given value after a successful comparison of one or more connection parameters against the regular expression. • Up to six **test** statements can be entered on the command line, and multiple **test** statements can be entered in the ruleset.
Step 9	**set** [*line-number*] {**pad** | **telnet**} *variable-parameter* [{**pad** | **telnet**} *variable-parameter* [...]] **Example:** `Router(cfg-pt-ruleset)# set telnet printer Y telnet binary Y`	Sets one or more connection parameters to a fixed value for a translation ruleset. • Once an incoming connection has been matched for processing, the ruleset generates the protocol translation parameters using a template that unconditionally sets a value defined by a **set** statement. • Up to six **set** statements can be entered on the command line, and multiple **set** statements can be entered in the ruleset.
Step 10	**options** *rule-option value* [*rule-option value* […]] **Example:** `Router(cfg-pt-ruleset)# options max-users 10 login`	Specifies protocol translation options in the translation ruleset. Choose from the following options for the *rule-option value* arguments: • **access-class** *number*—Defined access class number that the incoming connection must match. • **login**—Require login on the incoming connection (no value required). • **max-users** *number*—Maximum number of concurrent users allowed per ruleset. • **quiet**—Suppress translation information messages on the session (no value required).
Step 11	**exit** **Example:** `Router(cfg-pt-ruleset)# exit`	Exits the current configuration mode.

Testing and Maintaining Protocol Translation Rulesets

Perform this task to test and review your protocol translation rulesets.

SUMMARY STEPS

1. **enable**
2. **test translate** {**pad** | **telnet** | **parameter** *parameter*} [**detail**]
3. **show translate ruleset** [*name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**test translate** {**pad** | **telnet** | **parameter** *parameter*} [**detail**] **Example:** `Router# test translate pad detail`	Displays a trace of protocol translation behavior for a connection attempt. • **parameter**—Tests one of the translation ruleset parameters in interactive mode • **detail**—Displays an extended trace report about the configuration and connections.
Step 3	**show translate ruleset** [*name*] **Example:** `Router# show translate ruleset`	Displays a summary of a specific or of all configured translate rulesets and translate commands, behavioral parameters, and usage statistics. • The output of this command identifies **match**, **skip**, **set**, **test**, and **substitute** statement lines and numbers them; the line number can be used to reconfigure or remove any of these statements. • When the optional *name* argument is used, the display includes only details about the configured ruleset and does not include information about the one-line translate commands.

Configuration Examples for the Protocol Translation Ruleset Feature

This section provides the following examples:

PAD-to-Telnet Translation Ruleset: Example

In the following example, the incoming PAD address 55555 yields Telnet address 10.2.2.1, port 23 (default Telnet port). The local Boolean flag in the substitute statement specifies that Telnet protocol negotiations for PAD connections with destination addresses 55550 through 55555 should be forwarded, not processed.

```
translate ruleset P_to_T from pad to telnet
 description forwards control sequences
 match dest-addr ^5555.$
 set telnet dest-addr 10.2.2.1
 set telnet local n
 substitute pad dest-addr ^5555([0-5])$ into telnet local Y
 test telnet local n set telnet dest-port 2200
```

SVC Conversion with Translation Ruleset Service Selection: Example

The following example shows the selection of the outbound X.25 serial interface and the PAD profile for the Calling application based on the IP port number:

```
! define the profiles to be used by ruleset svc_service
x.29 profile ENG 2:0 3:128 4:0
x.29 profile DOC 2:0 3:128 4:0
x.29 profile MRKT 2:0 3:128 4:0
!
translate ruleset svc_service from telnet to pad
 match dest-addr ^10.10.1.6$ dest-port ^[1]00[0-1][0-8][1-3]$
 test telnet dest-port ^.0... set pad profile ENG
 test telnet dest-port ^.1... set pad profile DOC
 test telnet dest-port ^.2... set pad profile MRKT
 substitute telnet dest-port (.)$ into pad dest-addr 765432\1
 substitute telnet dest-port 0$ into pad dest-addr 76543210
```

Address Conversion in a Translation Ruleset: Example

The following translation ruleset example reduces the number of statements for converting the IP port number to an X.121 address for the following range of port numbers:

IP Address: 10.10.1.5 10000-19999 to X.121 Address 5559000000-9999
IP Address: 10.10.1.5 20000-29999 to X.121 Address 5559010000-9999
IP Address: 10.10.1.5 30000-39999 to X.121 Address 5559020000-9999
IP Address: 10.10.1.5 40000-49999 to X.121 Address 5559110000-9999
IP Address: 10.10.1.5 50000-59999 to X.121 Address 5559200000-9999

```
translate use telnet 10.10.1.5
!
translate ruleset T_to_P from telnet to pad
 description Site1 10.10.1.5 Area Code 555 exchgs 900, 901, 902, 911, 920
 match dest-addr ^10.10.1.5$ dest-port ^[1-5]....$
 substitute telnet dest-port ^1(....) into pad dest-addr 555900\1
 substitute telnet dest-port ^2(....) into pad dest-addr 555901\1
 substitute telnet dest-port ^3(....) into pad dest-addr 555902\1
 substitute telnet dest-port ^4(....) into pad dest-addr 555911\1
 substitute telnet dest-port ^5(....) into pad dest-addr 555920\1
```

Reserve PVC for Protocol Translation Ruleset: Example

The following example shows how to reserve a PVC for protocol translation ruleset handling, and select the outbound X.25 serial interface and PVC number based on the IP port number:

```
interface serial 2/0
x25 pvc 4 translate ruleset port_to_pvc
!
translate use telnet 10.10.1.6
!
translate ruleset port_to_pvc from telnet to pad
 match dest-addr ^10.10.1.6$ dest-port ^[12]00[0-7][1-3]$
 substitute telnet dest-port ^..0([0-7]) into pad interface serial 0/\1
 substitute telnet dest-port ^....(.) into pad pvc \1
 test telnet dest-port ^.0... set pad profile TEMS
 test telnet dest-port ^.1... set pad profile SQAS
 test telnet dest-port ^.2... set pad profile NMA
 substitute telnet dest-port (.)$ into pad dest-addr 876543\1

x.29 profile TEMS 2:0 3:128 4:0
x.29 profile SQAS 2:0 3:128 4:0
x.29 profile NMA 2:0 3:128 4:0
```

Displaying Ruleset Configuration Parameters: Example

The following example displays a summary of a configured translate ruleset named Template_1 that includes behavioral parameters, usage statistics, and line numbers for maintaining the configuration:

```
Router# show translate ruleset Template_1

PT ruleset Template_1, from telnet to pad
  administrative locks: 2 (2 readers, 0 writers)
  translations: 0 created, 0 active, 0 failed (0 max-user), 0 created for test
  match/skip lines: 1
    #1 match on 2 telnet tests: dest-addr ^172\.18\..*, dest-port ^12(0-7)..$
  options: login user, limited to 10 active sessions
  set/test/substitute lines: 3
    #1 set 2 parameters: telnet/printer Y, telnet/binary Y
    #2 set 1 parameter: pad/profile cust-profile-one
    #3 test 2 parameters: telnet/dest-addr ^172\.18\.0\.*, telnet/dest-port ^10.00; to set
2: pad/pvc 1, telnet/binary T
```

Testing the Ruleset Configuration Parameters: Example

The following example shows a detailed trace of PAD ruleset configurations:

```
Router# test translate pad detail

No PAD translate command matched
(Testing translate command A ...)
Ruleset A match/skip line 1 compared: match
  (processing set/test/substitute line 1)
  (set/test/subst line 1, item 1, parameter dest-addr set to 10.2.2.1)
  (processing set/test/substitute line 2)
  (set/test/subst line 2, item 1, parameter idle set to 10)
  (parsed pad parameter idle: 10)
  (parsed telnet parameter dest-addr: 10.2.2.1)
Ruleset A; pad parameter read:
  pad/dest-addr: 55555
```

```
Parameters set:
  pad/idle: 10
  telnet/dest-addr: 10.2.2.1
  (translation requires 0 bytes variable-sized memory)

Translate From: PAD 55555
                Idle 0:10
          To:   Telnet 10.2.2.1 Port 23
          Ruleset A
          0/1 users active
```

Additional References

The following sections provide references related to the Protocol Translation Ruleset feature.

Related Documents

Related Topic	Document Title
Regular expressions	Regular Expressions
Protocol translation	Configuring Protocol Translation and Virtual Asynchronous Devices

Standards

Standards	Title
None	—

MIBs

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

RFCs

RFCs	Title
None	—

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

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

- **description (ruleset)**
- **match (ruleset)**
- **options (ruleset)**
- **set (ruleset)**
- **show translate ruleset**
- **skip (ruleset)**
- **substitute (ruleset)**
- **test (ruleset)**
- **test translate**
- **translate ruleset**
- **translate use telnet**
- **x25 pvc translate ruleset**

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2008 Cisco Systems, Inc. All rights reserved.

Regular Expressions

First Published: February 4, 2004
Last Updated: April 15, 2011

This appendix explains regular expressions and how to use them in Cisco IOS software commands. It also provides details for composing regular expressions.

Contents

Information About Regular Expressions

Regular expressions are strings of special characters that can be used to search and find character patterns. A regular expression is entered as part of a command, and it is a pattern made up of symbols, letters, and numbers that represent an input string. Matching the string to the specified pattern is called *pattern matching*.

Pattern matching either succeeds or fails. If a regular expression can match two different parts of an input string, it will match the earliest part first.

Cisco configurations use regular expression pattern matching in several implementations. The following is a list of some of these implementations:

- Border Gateway Protocol (BGP)IP autonomous system path and X.29 access lists
- Modem (or chat) and system scripts
- X.25 route substitute destination feature
- Protocol translation ruleset scripts

Cisco Regular Expression Pattern Matching Characters

Table 1 summarizes the basic Cisco IOS regular expression characters and their functions.

Table 1 Cisco Regular Expression Characters

Regular Expression Character	Function	Examples
.	Matches any single character.	0.0 matches 0x0 and 020 t..t matches strings such as test, text, and tart
\	Matches the character following the backslash. Also matches (escapes) special characters.	172\.1\.. matches 172.1.10.10 but not 172.12.0.0 \. allows a period to be matched as a period
[ ]	Matches the characters or a range of characters separated by a hyphen, within left and right square brackets.	[02468a-z] matches 0, 4, and w, but not 1, 9, or K
^	Matches the character or null string at the beginning of an input string.	^123 matches 1234, but not 01234
?	Matches zero or one occurrence of the pattern. (Precede the question mark with Ctrl-V sequence to prevent it from being interpreted as a help command.)	ba?b matches bb and bab
$	Matches the character or null string at the end of an input string.	123$ matches 0123, but not 1234
*	Matches zero or more sequences of the character preceding the asterisk. Also acts as a wildcard for matching any number of characters.	5* matches any occurrence of the number 5 including none 18\..* matches the characters 18. and any characters that follow 18.
+	Matches one or more sequences of the character preceding the plus sign.	8+ requires there to be at least one number 8 in the string to be matched
() []	Nest characters for matching. Separate endpoints of a range with a dash (-).	(17)* matches any number of the two-character string 17 ([A-Za-z][0-9])+ matches one or more instances of letter-digit pairs, for example, b8 and W4

Table 1 Cisco Regular Expression Characters (continued)

Regular Expression Character	Function	Examples
|	Concatenates constructs. Matches one of the characters or character patterns on either side of the vertical bar.	A(B|C)D matches ABD and ACD, but not AD, ABCD, ABBD, or ACCD
_	Replaces a long regular expression list by matching a comma (,), left brace ({), right brace (}), the beginning of the input string, the end of the input string, or a space.	The characters _1300_ can match any of the following strings: ^1300$ ^1300space space1300 {1300, ,1300, {1300} ,1300,

The order for matching using the * or + character is longest construct first. Nested constructs are matched from the outside in. Concatenated constructs are matched beginning at the left side.

Single-Character Patterns

The simplest regular expression is a single character that matches itself in the input string. For example, the single-character regular expression 3 matches a corresponding 3 in the input string. You can use any letter (A to Z, a to z) or number (0 to 9) as a single-character pattern. You can also use a keyboard character other than a letter or a number, such as an exclamation point (!) or a tilde (~), as a single-character pattern, but not the characters listed in Table 1 that have special meaning when used in regular expressions.

To use the characters listed in Table 1 as single-character patterns, remove the special meaning by preceding each character with a backslash (\). The following examples are single-character patterns matching a dollar sign, an underscore, and a plus sign, respectively:

```
\$
\_
\+
```

You can specify a range of single-character patterns to match against a string. For example, you can create a regular expression that matches a string containing one of the following letters: a, e, i, o, and u. Only one of these characters must exist in the string for pattern matching to succeed. To specify a range of single-character patterns, enclose the single-character patterns in square brackets ([ ]). The order of characters within the brackets is not important. For example, [aeiou] matches any one of the five vowels of the lowercase alphabet, while [abcdABCD] matches any one of the first four letters of the lowercase or uppercase alphabet.

You can simplify ranges by typing only the endpoints of the range separated by a hyphen (-). Simplify the previous range as follows:

```
[a-dA-D]
```

To add a hyphen as a single-character pattern in your range, include another hyphen and precede it with a backslash:

[a-dA-D\-]

You can also include a right square bracket (]) as a single-character pattern in your range. To do so, enter the following:

[a-dA-D\-\]]

This example matches any one of the first four letters of the lowercase or uppercase alphabet, a hyphen, or a right square bracket.

You can reverse the matching of the range by including a caret (^) sign at the start of the range. The following example matches any letter except the ones listed:

[^a-dqsv]

The following example matches anything except a right square bracket (]) or the letter d:

[^\]d]

Multiple-Character Patterns

When creating regular expressions, you can also specify a pattern containing multiple characters. You create multiple-character regular expressions by joining letters, numbers, or keyboard characters that do not have special meaning. For example, a4% is a multiple-character regular expression. Precede keyboard characters that have special meaning with a backslash (\)when you want to remove their special meaning.

With multiple-character patterns, the order is important. The regular expression a4% matches the character a followed by the number 4 followed by a % sign. If the input string does not have a4%, in that order, pattern matching fails. The multiple-character regular expression a. uses the special meaning of the period character (.) to match the letter a followed by any single character. With this example, the strings ab, a!, and a2 are all valid matches for the regular expression.

You can create a multiple-character regular expressions containing all letters, all digits, all special keyboard characters, or a combination of letters, digits, and other keyboard characters.

Multipliers

You can create more complex regular expressions that instruct the Cisco IOS software to match multiple occurrences of a specified regular expression. To do so, you can use some special characters with your single- and multiple-character patterns.

The following example matches any number of occurrences of the letter a, including none:

a*

The following pattern requires that at least one letter a be present in the string to be matched:

a+

The following string matches any number of asterisks (*):

******

To use multipliers with multiple-character patterns, enclose the pattern in parentheses. In the following example, the pattern matches any number of the multiple-character string ab:

(ab)*

As a more complex example, the following pattern matches one or more instances of alphanumeric pairs (but not none; that is, an empty string is not a match):

([A-Za-z][0-9])+

Alternation

Alternation allows you to specify alternative patterns to match against a string. Separate the alternative patterns with a vertical bar (|). Exactly one of the alternatives can match the input string. For example, the regular expression codex|telebit matches the string codex or the string telebit, but not both codex and telebit.

Anchoring

You can instruct the Cisco IOS software to match a regular expression pattern against the beginning or the end of the input string. That is, you can specify that the beginning or end of an input string contain a specific pattern.

As an example, the following regular expression matches an input string only if the string starts with abcd:

^abcd

Whereas the following expression is a range that matches any single letter, as long as it is not the letters a, b, c, or d:

[^abcd]

With the following example, the regular expression matches an input string that ends with .12:

\.12$

Contrast these anchoring characters with the special character underscore (_). Underscore matches the beginning of a string (^), the end of a string ($), space ( ), braces ({ }), comma (,), or underscore (_). With the underscore character, you can specify a pattern to exist anywhere in the input string. For example, _1300_ matches any string that has 1300 somewhere in the string. The string's 1300 can be preceded by or end with a space, brace, comma, or underscore. So, while {1300_ matches the regular expression, 21300 and 13000 do not.

Parentheses for Recall

As shown in the "Multipliers" section, you can use parentheses with multiple-character regular expressions to multiply the occurrence of a pattern. You can also use parentheses around a single- or multiple-character pattern to instruct the Cisco IOS software to remember a pattern for use elsewhere in the regular expression.

To create a regular expression that recalls a previous pattern, use parentheses to instruct memory of a specific pattern and a backslash (\), followed by an integer to reuse the remembered pattern. The integer specifies the occurrence of a parentheses in the regular expression pattern. If you have more than one remembered pattern in your regular expression, then \1 uses the first remembered pattern, \2 uses the second remembered pattern, and so on.

The following regular expression uses parentheses for recall:

a(.)bc(.)\1\2

This regular expression matches the letter a, followed by any character (call it character #1), followed by bc, followed by any character (character #2), followed by character #1 again, followed by character #2 again. In this way, the regular expression can match aZbcTZT. The software identifies character #1 as Z and character #2 asT, and then uses Z and T again later in the regular expression.

The parentheses do not change the pattern; they only instruct the software to recall that part of the matched string. The regular expression (a)b still matches the input string ab, and (^3107) still matches a string beginning with 3107, but now the Cisco IOS software can recall the a of the ab string and the starting 3107 of another string for later use.

Examples of Regular Expressions

Example: Regular Expression Pattern Matching in Access Lists

Both the BGP IP autonomous system path feature and the X.29 access list configuration statements can use regular expression patterns to match addresses for allowing or denying access.

Within the scope of BGP in Cisco IOS software, regular expressions can be used in **show** commands and path filtering to match BGP prefixes based on the information contained in their autonomous system path.

Path filtering is defined with filters based on the autonomous system path:

```
ip as-path access-list acl-number [permit|deny] regexp
```

Additionally, based on BGP autonomous system paths, you can specify an access list filter on incoming updates from and outbound updates to neighbors by using filter lists or route maps.

Table 2 lists some commonly used regular expressions and their meanings.

Table 2 Commonly Used Regular Expressions

Expression	Meaning
.*	Anything
^$	Locally originated routes
^100_	Learned from autonomous system 100

Table 2 Commonly Used Regular Expressions

Expression	Meaning
_100$	Originated in autonomous system 100
_100_	Any instance of autonomous system 100
^[0-9]+$	Directly connected autonomous system paths

The following examples show regular expression pattern matching in access lists.

Accept Only Prefixes Originated in Autonomous System 5044

```
router bgp 65022
 no synchronization
 neighbor 172.16.0.1 remote-as 4
 neighbor 172.16.0.1 filter-list 1 in
 no auto-summary
!
ip as-path access-list 1 permit _5044$

Router# show ip bgp

BGP table version is 2, local router ID is 172.16.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.1     172.31.0.1         4 5044 i
```

Deny All Prefixes Originated in Autonomous System 200

```
router bgp 100
 neighbor 10.1.1.1 remote-as 65535
 neighbor 10.1.1.1 route-map map1 in
!
route-map map1 permit 10
 match as-path 1
!
ip as-path access-list 5 deny _200$
ip as-path access-list 5 permit .*
```

Accept Only Prefixes Announced via Autonomous System 5044

```
router bgp 65022
 no synchronization
 neighbor 172.17.0.1 remote-as 4
 neighbor 172.17.0.1 filter-list 1 in
 no auto-summary
!
ip as-path access-list 1 permit _5044_
```

Only Announce Routes Originated from Autonomous System 10

```
router bgp 100
 neighbor 10.1.1.1 remote-as 65535
 neighbor 10.1.1.1 route-map map1 out
!
route-map map1 permit 10
 match as-path 1
!
ip as-path access-list 1 permit ^$
```

Only Accept Routes Originated from a Locally Connected Autonomous System

```
router bgp 100
 neighbor 10.1.1.1 remote-as 65535
 neighbor 10.1.1.1 route-map map1 in
!
route-map map1 permit 10
 match as-path 1
!
ip as-path access-list 1 permit ^[0-9]+$
```

Only Accept Routes Originated from Autonomous System 10 or Contiguous to Autonomous System 10 (Example: [10, 254] but Not [10, 254, 333]

```
router bgp 100
 neighbor 10.1.1.1 remote-as 65535
 neighbor 10.1.1.1 route-map map1 in
!
route-map map1 permit 10
 match as-path 1
!
ip as-path access-list 22 permit ^10_[0-9]*$
```

Regular Expression ^123.*

The following example specifies that the BGP neighbor with IP address 172.23.1.1 is not sent advertisements about any path through or from the adjacent autonomous system 123:

```
ip as-path access-list 1 deny ^123 .*

router bgp 109
network 172.18.0.0
neighbor 172.19.6.6 remote-as 123
neighbor 172.23.1.1 remote-as 47
neighbor 10.125.1.1 filter-list 1 out
```

Regular Expression ^4$

The following example uses the regular expression string ^4$ to configure the router to receive the routes originated from only autonomous system 4:

```
ip as-path access-list 1 permit ^4$

router bgp 1
 neighbor 10.1.1.1 remote-as 4
 neighbor 10.1.1.1 route-map map1 in

route-map map1 permit 10
 match as-path 1
```

Testing the Regular Expression

Before you apply a regular expression through a filter list or a route map, you can test the output of this regular expression using the **show ip bgp regexp** command.

For example, to see any BGP prefix originated from one of your peers, type the following:

```
show ip bgp regexp ^[0-9]+$
```

X29 Access List

An X.29 access list can contain any number of access list items. The list items are processed in the order in which they are entered, with the first regular expression pattern match causing the permit or deny condition. The following example permits connections to hosts with addresses beginning with the string 31370:

```
x29 access-list 2 permit ^31370
```

Example: Regular Expression Pattern Matching in Scripts

On asynchronous lines, chat scripts send commands for modem dialing and logging in to remote systems. You can use a regular expression in the **script dialer** command to specify the name of the chat script that Cisco IOS software will execute on a particular asynchronous line.

You can also use regular expressions in the **dialer map** command to specify a modem script or system script to be used for a connection to one or multiple sites on an asynchronous interface.

The following example uses regular expressions telebit.* and usr.* to identify chat scripts for Telebit and US Robotics modems. When the chat script name (the string) matches the regular expression (the pattern specified in the command), then Cisco IOS software uses that chat script for the specified lines. For lines 1 and 6, Cisco IOS software uses the chat script named *telebit,* followed by any number of occurrences (*) of any character (.). For lines 7 and 12, the software uses the chat script named *usr,* followed by any number of occurrences (*) of any character (.).

```
! Some lines have Telebit modems.
line 1 6
chat-script telebit.*
! Some lines have US Robotics modems.
line 7 12
chat-script usr.*
```

If you adhere to a chat script naming convention of the form [modem-script *modulation-type] in the **dialer map** command, .*-v32bis for example, this allows you to specify the modulation type that is best for the system you are calling, and allows the modem type for the line to be specified by the modem **chat-script** command.

The following example shows the use of chat scripts implemented with the *system-script* and *modem-script* options of the **dialer map** command. If there is traffic for IP address 10.2.3.4, the router will dial the 91800 number using the *usrobotics-v32* script, matching the regular expression in the modem chat script. Then, the router will run the unix-slip chat script as the system script to log in. If there is traffic for 10.3.2.1, the router will dial 8899 using *usrobotics-v32*, matching both the modem script and modem chat script regular expressions. The router will then log in using the cisco-compressed script.

```
! Script for dialing a usr v.32 modem:
chat-script usrobotics-v32 ABORT ERROR "" "AT Z" OK "ATDT \T" TIMEOUT 60 CONNECT \c
!
! Script for logging into a UNIX system and starting up SLIP:
chat-script unix-slip ABORT invalid TIMEOUT 60 name: billw word: wewpass ">" "slip
default"
!
! Script for logging into a Cisco access server and starting up TCP header compression:
chat-script cisco-compressed...
!
line 15
 script dialer usrobotics-*
!
interface async 15
 dialer map ip 10.2.3.4 system-script *-v32 system-script cisco-compressed 91800
 dialer map ip 10.3.2.1 modem-script *-v32 modem-script cisco-compressed 91800
```

Example: Regular Expression Pattern Matching in X.25 Routing Entries

The **x25 route** command is used to create an entry in the X.25 routing table that the router consults to learn where to forward incoming calls and place outgoing packet assembler/disassembler (PAD) or protocol translation calls. Regular expressions are used with the **x25 route** command to allow pattern-matching operations on the addresses and user data. A common operation is to use prefix matching on the X.121 Data Network Identification Code (DNIC) field and route accordingly. The caret sign anchors the match to the beginning of the pattern.

In the following example, the **x25 route** command causes all X.25 calls to addresses whose first four DNIC digits are 1111 to be routed to serial interface 3. Note that the first four digits (^1111) are followed by a regular expression pattern that Cisco IOS software is to remember for later use. The \1 in the rewrite pattern recalls the portion of the original address matched by the digits following 1111, but changes the first four digits (1111) to 2222.

```
x25 route ^1111(.*) substitute-dest 2222\1 interface serial 3
```

The following example routes any incoming calls that begin with 2222 to the specified data-link connection identifier (DLCI) link:

```
x25 route ^2222 interface serial 1 dlci 20
```

The following example uses the regular expression ^ (carat) character to prevent (clear) X.25 routing for calls that do not specify a source address:

```
x25 route source ^$ clear
```

Example: Regular Expression Pattern Matching in a Protocol Translation Ruleset

Note Protocol translation rulesets are supported only in Cisco IOS Release 12.3(8)T and later releases.

Regular expressions for the Protocol Translation Ruleset feature have two uses: they match a text string against a defined pattern, and they use information from a defined regular expression match operation to create a different string using substitution. These operations are performed by combining the characters described in Table 1 with commands from translate ruleset configuration mode.

To understand regular expression pattern matching, begin by using Table 1 to interpret the following regular expression statement to match a string starting with the characters 172.18.:

^172\.18\..*

The following regular expression statement matches a five-digit number starting with 10 or 11:

^1[0-1]...$

Consider the following set of actions in a ruleset named B. This ruleset listens for incoming Telnet connections from a particular IP address and port number, but ignores (skips) others, decides which PAD destination address the matched incoming connections should be connected to, and finally sets the PAD connection's X.25 virtual circuit (VC) idle timer from the first digit of the port number.

```
translate ruleset B from telnet to pad
match dest-addr ^10.2.2.(..)$ dest-port ^20..$
skip dest-addr ^10.2.2.11$
set pad dest-addr 4444
substitute telnet dest-port ^200(.)$ into pad idle \1
```

The caret sign anchors a match to the beginning of a string, in this example, 10.2.2 for the destination address and 20 for the destination port.

The parentheses are a powerful tool for the regular expression match operation because they identify groups of characters needed for a substitution. Combined with the “substitute...into” statement, the parentheses can dynamically create a broad range of string patterns and connection configurations.

In the example, the periods in the parentheses pair can be thought of as placeholders for the characters to be substituted. The dollar sign anchors the substitution match to the end of a string. The backslash preceding the number makes it a literal setting, so no substitution will be done to the idle timer setting.

The **test translate ruleset** command tests the script, and for the previous example, the command would provide a report like the following:

```
Translate From: Telnet 10.2.2.10 Port 2000
          To:   PAD 4444
          Ruleset B
          0/1 users active
```

Consider the following more complex expression:

^172\.18\.(10)\.(.*)$.

This expression matches any string beginning with 172.18. and identifies two groups, one that matches 10 and the other that matches a wildcard character.

Let us say that the regular expression ^172\.18\.(10)\.(.*)$ matched the characters 172.18.10.255 from an incoming connection. Once the match is made, the software places the character groups 10 and 255 into buffers and writes the matched groups using a substitution expression.

Regular expression substitution into the expression 0001172018\1\2 would generate the string 000117201810255.

The regular expression \0 would write the entire matched string and substitution into the expression 0001\0, and would generate the string 0001172.18.10.255.

Additional References

Related Documents

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

Technical Assistance

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

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© 20018 Cisco Systems, Inc. All rights reserved

X.3 PAD Parameters

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. That is, it can take data packets from a host and return them into a character stream that can be sent to the terminals, or start-stop mode DTE, as defined by the International Telecommunication Union (ITU). A PAD is defined by ITU-T Recommendations X.3, X.28, and X.29. (The ITU-T carries out the functions of the former Consultative Committee for International Telegraph and Telephone.)

ITU-T Recommendation X.3 specifies the parameters for terminal-handling functions such as data speed, flow control, character echoing, and other functions for a connection to an X.25 host. The X.3 parameters are similar in function to the Telnet options.

ITU-T 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 the request, 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 ITU-T also provides Recommendation X.28 to specify the user interface for locally controlling a PAD.

Cisco IOS software offers two ways of connecting to a PAD: using the **pad** EXEC user interface command to initiate an outgoing connection to a PAD, and using the **x28** EXEC command to access the Cisco universal X.28 PAD user emulation mode.

In X.28 PAD user emulation mode, you can perform the same functions available from the Cisco **pad** EXEC user interface; however, X.28 PAD user emulation mode adds functionality such as the ability to exchange PAD signals across an X.25 network, and is useful for connecting to systems using software designed to interact with an X.28 PAD. X.28 PAD user emulation mode is also useful when a reverse connection requires packetization according to the X.29 parameters.

This appendix discusses the X.3 PAD parameters. The chapter "Configuring the Cisco PAD Facility for X.25 Connections" in this publication explains how to make PAD connections and how to switch between connections. Refer to the ITU-T X.3 and X.28 recommendations for additional information about the X.3 PAD parameters.

X.3 PAD Parameter Descriptions

Following are descriptions of X.3 parameters 1 through 22. 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 access server sends an X.29 SET PARAMETER packet to set the noted defaults.

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

Note Some of the commands described in this section require ASCII decimal values. Refer to the "ASCII Character Set and Hex Values" appendix in the *Cisco IOS Configuration Fundamentals Command Reference*, Release 12.2, for a list of ASCII characters. Also note that the PAD EXEC user interface and X.28 PAD user emulation mode provide different support for the PAD parameters, and these differences are noted in the following descriptions.

Parameter 1: PAD Recall Using a Character

Parameter 1 determines whether the start-stop mode DTE is allowed to escape from data transfer mode to send PAD command signals.

Because the PAD EXEC mode 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. The PAD EXEC user interface does not support this parameter; however, the Cisco X.28 standard user interface does support this parameter.

Minimum value: 0; maximum value: 126; X.28 PAD user emulation mode default: 1.

Parameter 2: Echo

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 off local echo on the remote PAD to support the Cisco user interface. See Table 1 for local echo mode values and their descriptions.

Table 1 PAD Local Echo Mode Values

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

Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode defaults: 1.

Parameter 3: Selection of Data Forwarding 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. See Table 2 for data forward character values and their descriptions.

Table 2 PAD Data Forward Character Values

Value	Description
0	None—full packet.
1	Forward packet upon receipt of an alphanumeric character.
2	Forward packet upon receipt of an ASCII CR (a Return is the outgoing connection default).
4	Forward packet upon receipt of an ASCII ESCAPE, BEL, ENQ, or ACK.
8	Forward packet upon receipt of an ASCII DEL, CAN, or DC2.
16	Forward packet upon receipt of an ASCII ETX or EOT.
32	Forward packet upon receipt of an ASCII HT, LT, VT, or FF.
64	All other characters in columns 0 and 1 of the ASCII chart not listed.
128	Forward packet upon receipt of a semicolon (;).

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

A forwarding mask can also be statically set using the **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.

Minimum value: 0; maximum value: 255; PAD EXEC mode default: 2 (ASCII CR); X.28 PAD user emulation mode default: 126 (ASCII ~).

Parameter 4: Selection of Idle Timer Delay

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. See Table 3 for PAD idle timer values and their descriptions.

Table 3 PAD Idle Timer Values

Value	Description
0	No timer.
1–255	Delay value in twentieths of a second (default for both connection types is 1).

Minimum value: 0; maximum value: 255; PAD EXEC mode default: 1; X.28 PAD user emulation mode default: 0.

Parameter 5: Ancillary Device Control

Parameter 5 selects whether PAD can send flow control X-ON/X-OFF (ASCII DC1/DC3 transmission on and off) characters during data transfer to the start-stop mode DTE to control the terminal and data flow. Flow control is not directly supported on access servers 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.

See Table 4 for PAD flow control signal values and their descriptions.

Table 4 PAD Flow Control Signal Values

Value	Description
0	No use of X-ON/X-OFF.
1	Use of X-ON/X-OFF (data transfer).
2	Use of X-ON/X-OFF (data transfer and command).

Minimum value: 0; maximum value: 2; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 1.

Parameter 6: Control of PAD Service Signals

Parameter 6 controls PAD service signals and the prompt. By default, the Cisco X.28 standard user interface prompt is an asterisk (*), but the prompt can be changed. See Table 5 for PAD BREAK signal values and their descriptions.

Table 5 PAD BREAK Service Signal Values

Value	Description
0	No service signals are sent to the start-stop DTE.
1	Service signals other than the prompt PAD service signal are sent.
2	Editing PAD service signals are only sent in the format specified by parameter 19.
4	The prompt PAD service signal is sent in the standard format.
8 to 15	PAD service signals are only sent in network-dependent format. Value 8 specifies the prompt as x28>. Value 9 enables French extended mode support. Value 10 specifies the prompt be the same as the Cisco EXEC prompt.

The PAD EXEC user interface does not support this parameter; however, the Cisco X.28 standard user interface does support this parameter.

Minimum value: 0; maximum value: 255; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 2.

Parameter 7: Selection of Operation of PAD on Receipt of a BREAK Signal

Parameter 7 defines the action of the PAD after receiving a BREAK signal from the from the start-stop mode DTE. See Table 6 for PAD BREAK signal values and their descriptions.

Table 6 PAD BREAK Signal Values

Value	Description
0	Ignore the BREAK signal.
1	Send an interrupt packet to notify the remote host or another PAD that the BREAK signal has been generated.
2	Send a Reset packet to reset the virtual circuit.
4	Send an X.29 Indication of Break to the remote host, or to a PAD (outgoing connection default).
8	Escape from data transfer mode.
16	Discard output to the start-stop mode DTE 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 X.29 Indication of Break 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 are logical. Common options are to begin discarding output and send both an X.25 Interrupt packet and an X.29 Indication of Break packet; these options are supported. All other options are not supported and are silently ignored.

Minimum value: 0; maximum value: 31; PAD EXEC mode default: 4; X.28 PAD user emulation mode default: 2.

Parameter 8: Discard Output

Parameter 8 indicates to the PAD whether to discard received packets rather than disassemble and send 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 0 restores normal data delivery to the terminal.

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

See Table 7 for PAD discard output values and their descriptions.

Table 7 PAD Discard Output Values

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

Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 9: Padding After Return

Parameter 9 determines whether PAD can provide padding (insert filler characters) upon receipt of an ASCII CR (Return) control code from the start-stop mode DTE.

Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 10: Line Folding (Not Supported)

Neither the PAD EXEC user interface nor the X.28 PAD user emulation mode supports this parameter.

Parameter 11: DTE Speed

Parameter 11 is a read-only value that determines the binary speed of the start-stop mode DTE sent across the interface between PAD and the access server. See Table 8 for PAD speed values and their descriptions.

Table 8 PAD DTE Speed 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

Minimum value: 0; maximum value: 18; PAD EXEC mode and X.28 PAD user emulation mode default: 14.

Parameter 12: Flow Control of the PAD by the Start-Stop Mode DTE

Parameter 12 determines whether the start-stop mode DTE can send ASCII X-ON/X-OFF characters to PAD during the data transfer mode. Flow control is not directly supported on access servers 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.

See Table 9 for PAD flow control values and their descriptions.

Table 9 PAD Flow Control Values

Value	Description
0	No use of X-ON/X-OFF.
1	Use of X-ON/X-OFF.

Minimum value: 0; maximum value: 1; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 1.

Parameter 13: Line Feed Insertion

Parameter 13 determines the procedure for inserting the line feed character upon receipt of an ASCII CR character. The PAD also responds to a value that results from the addition of any of the line feed signal values described in Table 10.

Table 10 PAD Line Feed Signal Values

Value	Description
0	Do not insert the line feed character (outgoing connection default).
1	Insert a line feed after sending an ASCII CR to the start-stop mode DTE.
2	Insert a line feed after echoing an ASCII CR to the start-stop mode DTE.
4	Insert a line feed after echoing an ASCII CR to the remote host.

Minimum value: 0; maximum value: 7; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 14: Line Feed Padding

Parameter 14 determines whether PAD can provide padding (insert filler characters) upon receipt of a line feed character from the start-stop mode DTE. This function is generally provided by the end-user operating system.

Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 15: Editing

Parameter 15 enables or disables a PAD editing function for the start-stop mode DTE in data transfer mode.

Enabling the editing function disables the idle timer (see parameter 4). The user at the start-stop mode DTE can make corrections and display the line buffer containing the characters to be sent when the data forwarding character (see parameter 3) is received. See Table 11 for PAD local editing function values and their descriptions.

Table 11 PAD Local Editing Functions

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

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

Minimum value: 0; maximum value: 1; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 16: Character Delete

Parameter 16 allows you to select a character that will delete a character while in PAD editing mode. This character is valid only if parameter 15 is set to 1. Select one character from the ASCII character set to represent the delete character.

Minimum value: 0; maximum value: 127; PAD EXEC mode and X.28 PAD user emulation mode default: 127 (ASCII 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 1. Select one character from the ASCII character set to represent the line delete character.

Minimum value: 0; maximum value: 127; PAD EXEC mode default: 21 (ASCII NAK or Ctrl-U); X.28 PAD user emulation mode default: 24 (ASCII CAN or Ctrl-X).

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 1. Select one character from the ASCII character set to represent the delete character.

Minimum value: 0; maximum value: 127; PAD EXEC mode and X.28 PAD user emulation mode default: 18 (ASCII DC2 or Ctrl-R).

Parameter 19: Editing PAD Service Signals

Parameter 19 allows you to set editing PAD service signals.

The PAD EXEC user interface does not support this parameter; however, the X.28 PAD user emulation mode does support this parameter.

See Table 12 for editing PAD service signal values and their descriptions.

Table 12 Editing PAD Service Signal Values

Value	Description
0	No editing PAD service signals.
1	Editing PAD service signals for printing terminals.
2	Editing PAD service signals for display terminals.
8; 32–126	Editing PAD service signals using an ASCII character in the value range.

Minimum value: 0; maximum value: 126; PAD EXEC mode default: 0; X.28 PAD user emulation mode default: 2.

Parameter 20: Echo Mask

Parameter 20 allows you to set the start-stop mode DTE to echo all characters.

The PAD EXEC user interface does not support this parameter; however, the X.28 PAD user emulation mode does support this parameter.

See Table 13 for PAD echo mask values and their descriptions.

Table 13 PAD Echo Mask Values

Value	Description
0	No echo mask (all characters echoed).
1	No echo of ASCII character CR.
2	No echo of ASCII character LF.
4	No echo of ASCII characters VT, HT, FF.
8	No echo of ASCII characters BEL or BS.
16	No echo of ASCII characters ESCAPE or ENQ.
32	No echo of ASCII characters ACK, NAK, STX, SOH, EOT, ETB, or ETX.
64	No echo of characters as designated by parameters 16, 17, or 18.
128	No echo of all other characters not listed and of ASCII DEL.

Minimum value: 0; maximum value: 255; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Parameter 21: Parity Treatment

Parameter 21 controls the parity and character format used by the start-stop mode DTE. Table 14 provides a brief description for each value of parameter 21. For a description of Cisco IOS X.28 default behavior for parameter 21, see Table 15; see Table 16 for a description of the Cisco IOS X.28 implementation for the French Transpac public switched data network and its technical specification and utilization of networks standards (STUR).

Table 14 Parity Treatment Values

Value	Description
0	No parity checking or generation (default). When the PAD transfers a data character or interprets a received character for a specific action different from the transfer of this data character to the remote DTE, it inspects only the first seven bits and will not take account of the eighth bit.
1	Check character parity against the parity configured on the asynchronous line, and drop character if invalid parity is set. **Note** The PAD treats the eighth bit of the characters received from the start-stop DTE as a parity bit and checks this bit against the type of parity used between the PAD and the start-stop mode DTE.
2	Generate parity. **Note** The PAD replaces the eighth bit of the characters to be sent to the start-stop mode DTE with the bit that corresponds to the type of parity used between the PAD and the start-stop mode DTE.
3	Check and generate parity (combination of 1 and 2). **Note** The PAD will both check the parity bit for characters received from the start-stop mode DTE and generate the parity bit for characters to be sent to the start-stop mode DTE, as described for values 1 and 2.
4	Pass parity transparently. **Note** The PAD transparently passes the eighth bit whenever it must transfer a data character or interpret a received character.

Minimum value: 0; maximum value: 4; PAD EXEC mode and X.28 PAD user emulation mode default: 0.

Table 15 Parity Treatment (Parameter 21) Cisco IOS and X.28 Standards Behavior

Selectable Values:	0	1	2	3	4
Direction	**No parity checking or generation**	**Parity checking**	**Parity generation**	**Parity checking and generation**	**No parity, transparent bit 8**
Terminal -> Host data	If parity, strip parity bit and send to host. If local echo, do local echo of character received.	Check 8th bit received against type of parity configured on the line. If parity error, discard the character and do not echo the character in error, and send the ASCII BELL signal to the terminal. Strip parity bit and send to host. If local echo, do local echo of character received.	Strip parity bit and send to host. If local echo and if parity, replace the 8th bit with type of parity configured on the line. Otherwise, do local echo of character received.	Check 8th bit received against type of parity configured on the line. If parity error, discard the character and do not echo the character in error, and send the ASCII BELL signal to the terminal. Strip parity bit and send to host. If local echo and if parity, replace the 8th bit with type of parity configured on the line. Otherwise, do local echo of character received.	Transparently pass the 8th bit. If local echo, do local echo of character received.
Host -> Terminal data	Transparently pass the 8th bit.	Transparently pass the 8th bit.	Replace 8th bit to be sent with the type of parity bit configured on the line.	Replace 8th bit to be sent with the type of parity bit configured on the line.	Transparently pass the 8th bit.
PAD generated data	Transmit with the type of parity configured on the line.[1]	Transmit with the type of parity configured on the line.	Transmit with the type of parity configured on the line.	Transmit with the type of parity configured on the line.	Transmit transparently.
Terminal -> PAD	Strip parity before testing the command string. Echo the character received.	Strip parity before testing the command string.[2] Echo with the type of parity configured on the line.	Strip parity before testing the command string. Echo with the type of parity configured on the line.	Strip parity before testing the command string.[2] Echo with the type of parity configured on the line.	Strip parity before testing the command string. Echo the character received.
CUD, terminal -> PAD	Pass all data bits; 8th bit is 0 in case of 7 data bits.	Pass all data bits; 8th bit is 0 in case of 7 data bits.	Pass all data bits; 8th bit is 0 in case of 7 data bits.	Pass all data bits; 8th bit is 0 in case of 7 data bits.	Pass all data bits; 8th bit is 0 in case of 7 data bits.[3]

1. Deviates from X.28 standards, should transmit with even parity.
2. Deviates from X.28 standards, should check parity against type of parity configured on the line.
3. Deviates from X.28 standards, should transparently pass all data bits.

Table 16 Parity Treatment (Parameter 21) Transpac/STUR Specific Behavior

Selectable Values:	0	1	2	3	4
Direction	**No parity checking or generation**	**Parity checking**	**Parity generation**	**Parity checking and generation**	**No parity, transparent bit 8**
Terminal -> Host data	Transparently pass the 8th bit.[1] If local echo, do local echo of character received.	Check the 8th bit received against type of parity configured on the line. If parity error, discard character and do not echo the character in error, and send the ASCII BELL signal to the terminal. Strip parity bit and send to host. If local echo, do local echo of character received.	Strip parity bit and send to host. If local echo and if parity, replace the 8th bit with type of parity configured on the line. Otherwise, do local echo of character received.	Check the 8th bit received against type of parity configured on the line. If parity error, discard character and do not echo the character in error, and send the ASCII BELL signal to the terminal. Strip parity bit and send to host. If local echo and if parity, replace the 8th bit with type of parity configured on the line. Otherwise, do local echo of character received.	Transparently pass the 8th bit. If local echo, do local echo of character received.
Host -> Terminal data	Transparently pass the 8th bit.	Transparently pass the 8th bit.	Replace the 8th bit to be sent with the type of parity bit configured on the line.	Replace the 8th bit to be sent with the type of parity bit configured on the line.	Transparently pass the 8th bit.
PAD generated data	Transmit with even parity.[2]	Transmit with the type of parity configured on the line.	Transmit with the type of parity configured on the line.	Transmit with the type of parity configured on the line.	Transmit transparently.
Terminal -> PAD	Strip parity before testing the command string. Echo the character received.	Strip parity before testing the command string.[3] Echo with the type of parity configured on the line.	Strip parity before testing the command string. Echo with the type of parity configured on the line.	Strip parity before testing the command string.[3] Echo with the type of parity configured on the line.	Strip parity before testing the command string. Echo the character received.
CUD, terminal -> PAD	Transparently pass all data bits.[1]	Pass all data bits with parity configured on the line.[1]	Pass all data bits; 8th bit is 0 in case of 7 data bits.	Pass all data bits with parity configured on the line.[1]	Transparently pass all data bits.[2]

1. Conforms to the Transpac and STUR specifications but not the X.28 standard.
2. Conforms to both the Transpac and STUR specifications and the X.28 standard.
3. Deviates from X.28 standards; should check parity against the type of parity configured on the line.

Parameter 22: Page Wait (Not Supported)

Neither the PAD EXEC user interface nor the X.28 PAD user emulation mode supports this parameter.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2008 Cisco Systems, Inc. All rights reserved.