Cisco IOS Terminal Services Configuration Guide

Release 12.2SX

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

Cisco IOS Terminal Services Configuration Guide

About Cisco IOS and Cisco IOS XE Software Documentation

Last updated: August 6, 2008

This document describes the objectives, audience, conventions, and organization used in Cisco IOS and Cisco IOS XE software documentation, collectively referred to in this document as Cisco IOS documentation. Also included are resources for obtaining technical assistance, additional documentation, and other information from Cisco. This document is organized into the following sections:

Documentation Objectives

Cisco IOS documentation describes the tasks and commands available to configure and maintain Cisco networking devices.

Audience

The Cisco IOS documentation set is i ntended for users who configure and maintain Cisco networking devices (such as routers and switches) but who may not be familiar with the configuration and maintenance tasks, the relationship among tasks, or the Cisco IOS commands necessary to perform particular tasks. The Cisco IOS documentation set is also intended for those users experienced with Cisco IOS who need to know about new features, new configuration options, and new software characteristics in the current Cisco IOS release.

Documentation Conventions

In Cisco IOS documentation, the term *router* may be used to refer to various Cisco products; for example, routers, access servers, and switches. These and other networking devices that support Cisco IOS software are shown interchangeably in examples and are used only for illustrative purposes. An example that shows one product does not necessarily mean that other products are not supported.

This section includes the following topics:

Typographic Conventions

Cisco IOS documentation uses the following typographic conventions:

Convention	Description
^ or Ctrl	Both the ^ symbol and Ctrl represent the Control (Ctrl) key on a keyboard. For example, the key combination **^D** or **Ctrl-D** means that you hold down the Control key while you press the D key. (Keys are indicated in capital letters but are not case sensitive.)
string	A string is a nonquoted set of characters shown in italics. For example, when setting a Simple Network Management Protocol (SNMP) community string to *public*, do not use quotation marks around the string; otherwise, the string will include the quotation marks.

Command Syntax Conventions

Cisco IOS documentation uses the following command syntax conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter as shown.
italic	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional keyword or argument.
|	A vertical line, called a pipe, indicates a choice within a set of keywords or arguments.
[x | y]	Square brackets enclosing keywords or arguments separated by a pipe indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a pipe indicate a required choice.
[x {y | z}]	Braces and a pipe within square brackets indicate a required choice within an optional element.

Software Conventions

Cisco IOS uses the following program code conventions:

Convention	Description
`Courier font`	Courier font is used for information that is displayed on a PC or terminal screen.
`Bold Courier font`	Bold Courier font indicates text that the user must enter.
< >	Angle brackets enclose text that is not displayed, such as a password. Angle brackets also are used in contexts in which the italic font style is not supported; for example, ASCII text.
!	An exclamation point at the beginning of a line indicates that the text that follows is a comment, not a line of code. An exclamation point is also displayed by Cisco IOS software for certain processes.
[ ]	Square brackets enclose default responses to system prompts.

Reader Alert Conventions

The Cisco IOS documentation set uses the following conventions for reader alerts:

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Documentation Organization

This section describes the Cisco IOS documentation set, how it is organized, and how to access it on Cisco.com. Included are lists of configuration guides, command references, and supplementary references and resources that make up the documentation set. The following topics are included:

Cisco IOS Documentation Set

Cisco IOS documentation consists of the following:

- Release notes and caveats provide information about platform, technology, and feature support for a release and describe severity 1 (catastrophic), severity 2 (severe), and severity 3 (moderate) defects in released Cisco IOS code. Review release notes before other documents to learn whether or not updates have been made to a feature.
- Sets of configuration guides and command references organized by technology and published for each standard Cisco IOS release.
  - Configuration guides—Compilations of documents that provide informational and task-oriented descriptions of Cisco IOS features.
  - Command references—Compilations of command pages that provide detailed information about the commands used in the Cisco IOS features and processes that make up the related configuration guides. For each technology, there is a single command reference that covers all Cisco IOS releases and that is updated at each standard release.
- Lists of all the commands in a specific release and all commands that are new, modified, removed, or replaced in the release.
- Command reference book for **debug** commands. Command pages are listed in alphabetical order.
- Reference book for system messages for all Cisco IOS releases.

Cisco IOS Documentation on Cisco.com

The following sections describe the documentation organization and how to access various document types.

Use Cisco Feature Navigator to find information about platform support and Cisco IOS and Catalyst OS software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

New Features List

The New Features List for each release provides a list of all features in the release with hyperlinks to the feature guides in which they are documented.

Feature Guides

Cisco IOS features are documented in feature guides. Feature guides describe one feature or a group of related features that are supported on many different software releases and platforms. Your Cisco IOS software release or platform may not support all the features documented in a feature guide. See the Feature Information table at the end of the feature guide for information about which features in that guide are supported in your software release.

Configuration Guides

Configuration guides are provided by technology and release and comprise a set of individual feature guides relevant to the release and technology.

Command References

Command reference books describe Cisco IOS commands that are supported in many different software releases and on many different platforms. The books are provided by technology. For information about all Cisco IOS commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or the *Cisco IOS Master Command List, All Releases*, at http://www.cisco.com/en/US/docs/ios/mcl/all_release/all_mcl.html.

Cisco IOS Supplementary Documents and Resources

Supplementary documents and resources are listed in Table 2 on page xi.

Configuration Guides, Command References, and Supplementary Resources

Table 1 lists, in alphabetical order, Cisco IOS and Cisco IOS XE software configuration guides and command references, including brief descriptions of the contents of the documents. The Cisco IOS command references are comprehensive, meaning that they include commands for both Cisco IOS software and Cisco IOS XE software, for all releases. The configuration guides and command references support many different software releases and platforms. Your Cisco IOS software release or platform may not support all these technologies.

For additional information about configuring and operating specific networking devices, go to the Product Support area of Cisco.com at http://www.cisco.com/web/psa/products/index.html.

Table 2 lists documents and resources that supplement the Cisco IOS software configuration guides and command references. These supplementary resources include release notes and caveats; master command lists; new, modified, removed, and replaced command lists; system messages; and the debug command reference.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS AppleTalk Configuration Guide *Cisco IOS XE AppleTalk Configuration Guide* *Cisco IOS AppleTalk Command Reference*	AppleTalk protocol.
Cisco IOS Asynchronous Transfer Mode Configuration Guide *Cisco IOS Asynchronous Transfer Mode Command Reference*	LAN ATM, multiprotocol over ATM (MPoA), and WAN ATM.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Bridging and IBM Networking Configuration Guide *Cisco IOS Bridging Command Reference* *Cisco IOS IBM Networking Command Reference*	• Transparent and source-route transparent (SRT) bridging, source-route bridging (SRB), Token Ring Inter-Switch Link (TRISL), and token ring route switch module (TRRSM). • Data-link switching plus (DLSw+), serial tunnel (STUN), block serial tunnel (BSTUN); logical link control, type 2 (LLC2), synchronous data link control (SDLC); IBM Network Media Translation, including Synchronous Data Logical Link Control (SDLLC) and qualified LLC (QLLC); downstream physical unit (DSPU), Systems Network Architecture (SNA) service point, SNA frame relay access, advanced peer-to-peer networking (APPN), native client interface architecture (NCIA) client/server topologies, and IBM Channel Attach.
Cisco IOS Broadband and DSL Configuration Guide *Cisco IOS XE Broadband and DSL Configuration Guide* *Cisco IOS Broadband and DSL Command Reference*	Point-to-Point Protocol (PPP) over ATM (PPPoA) and PPP over Ethernet (PPPoE).
Cisco IOS Carrier Ethernet Configuration Guide *Cisco IOS Carrier Ethernet Command Reference*	Connectivity fault management (CFM), Ethernet Local Management Interface (ELMI), IEEE 802.3ad link bundling, Link Layer Discovery Protocol (LLDP), media endpoint discovery (MED), and operations, administration, and maintenance (OAM).
Cisco IOS Configuration Fundamentals Configuration Guide *Cisco IOS XE Configuration Fundamentals Configuration Guide* *Cisco IOS Configuration Fundamentals Command Reference*	Autoinstall, Setup, Cisco IOS command-line interface (CLI), Cisco IOS file system (IFS), Cisco IOS web browser user interface (UI), basic file transfer services, and file management.
Cisco IOS DECnet Configuration Guide *Cisco IOS XE DECnet Configuration Guide* *Cisco IOS DECnet Command Reference*	DECnet protocol.
Cisco IOS Dial Technologies Configuration Guide *Cisco IOS XE Dial Technologies Configuration Guide* *Cisco IOS Dial Technologies Command Reference*	Asynchronous communications, dial backup, dialer technology, dial-in terminal services and AppleTalk remote access (ARA), large scale dialout, dial-on-demand routing, dialout, modem and resource pooling, ISDN, multilink PPP (MLP), PPP, virtual private dialup network (VPDN).
Cisco IOS Flexible NetFlow Configuration Guide *Cisco IOS Flexible NetFlow Command Reference*	Flexible NetFlow.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS H.323 Configuration Guide	Gatekeeper enhancements for managed voice services, Gatekeeper Transaction Message Protocol, gateway codec order preservation and shutdown control, H.323 dual tone multifrequency relay, H.323 version 2 enhancements, Network Address Translation (NAT) support of H.323 v2 Registration, Admission, and Status (RAS) protocol, tokenless call authorization, and VoIP gateway trunk and carrier-based routing.
Cisco IOS High Availability Configuration Guide *Cisco IOS XE High Availability Configuration Guide* *Cisco IOS High Availability Command Reference*	A variety of High Availability (HA) features and technologies that are available for different network segments (from enterprise access to service provider core) to facilitate creation of end-to-end highly available networks. Cisco IOS HA features and technologies can be categorized in three key areas: system-level resiliency, network-level resiliency, and embedded management for resiliency.
Cisco IOS Integrated Session Border Controller Command Reference	A VoIP-enabled device that is deployed at the edge of networks. An SBC is a toolkit of functions, such as signaling interworking, network hiding, security, and quality of service (QoS).
Cisco IOS Intelligent Service Gateway Configuration Guide *Cisco IOS Intelligent Service Gateway Command Reference*	Subscriber identification, service and policy determination, session creation, session policy enforcement, session life-cycle management, accounting for access and service usage, session state monitoring.
Cisco IOS Interface and Hardware Component Configuration Guide *Cisco IOS XE Interface and Hardware Component Configuration Guide* *Cisco IOS Interface and Hardware Component Command Reference*	LAN interfaces, logical interfaces, serial interfaces, virtual interfaces, and interface configuration.
Cisco IOS IP Addressing Services Configuration Guide *Cisco IOS XE Addressing Services Configuration Guide* *Cisco IOS IP Addressing Services Command Reference*	Address Resolution Protocol (ARP), Network Address Translation (NAT), Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Next Hop Address Resolution Protocol (NHRP).
Cisco IOS IP Application Services Configuration Guide *Cisco IOS XE IP Application Services Configuration Guide* *Cisco IOS IP Application Services Command Reference*	Enhanced Object Tracking (EOT), Gateway Load Balancing Protocol (GLBP), Hot Standby Router Protocol (HSRP), IP Services, Server Load Balancing (SLB), Stream Control Transmission Protocol (SCTP), TCP, Web Cache Communication Protocol (WCCP), User Datagram Protocol (UDP), and Virtual Router Redundancy Protocol (VRRP).
Cisco IOS IP Mobility Configuration Guide *Cisco IOS IP Mobility Command Reference*	Mobile ad hoc networks (MANet) and Cisco mobile networks.
Cisco IOS IP Multicast Configuration Guide *Cisco IOS XE IP Multicast Configuration Guide* *Cisco IOS IP Multicast Command Reference*	Protocol Independent Multicast (PIM) sparse mode (PIM-SM), bidirectional PIM (bidir-PIM), Source Specific Multicast (SSM), Multicast Source Discovery Protocol (MSDP), Internet Group Management Protocol (IGMP), and Multicast VPN (MVPN).

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS IP Routing Protocols Configuration Guide *Cisco IOS XE IP Routing Protocols Configuration Guide* *Cisco IOS IP Routing Protocols Command Reference*	Border Gateway Protocol (BGP), multiprotocol BGP, multiprotocol BGP extensions for IP multicast, bidirectional forwarding detection (BFD), Enhanced Interior Gateway Routing Protocol (EIGRP), Interior Gateway Routing Protocol (IGRP), Intermediate System-to-Intermediate System (IS-IS), on-demand routing (ODR), Open Shortest Path First (OSPF), and Routing Information Protocol (RIP).
Cisco IOS IP SLAs Configuration Guide *Cisco IOS XE IP SLAs Configuration Guide* *Cisco IOS IP SLAs Command Reference*	Cisco IOS IP Service Level Agreements (IP SLAs).
Cisco IOS IP Switching Configuration Guide *Cisco IOS XE IP Switching Configuration Guide* *Cisco IOS IP Switching Command Reference*	Cisco Express Forwarding, fast switching, and Multicast Distributed Switching (MDS).
Cisco IOS IPv6 Configuration Guide *Cisco IOS XE IPv6 Configuration Guide* *Cisco IOS IPv6 Command Reference*	For IPv6 features, protocols, and technologies, go to the IPv6 "Start Here" document at the following URL: http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-roadmap.html
Cisco IOS ISO CLNS Configuration Guide *Cisco IOS XE ISO CLNS Configuration Guide* *Cisco IOS ISO CLNS Command Reference*	ISO connectionless network service (CLNS).
Cisco IOS LAN Switching Configuration Guide *Cisco IOS XE LAN Switching Configuration Guide* *Cisco IOS LAN Switching Command Reference*	VLANs, Inter-Switch Link (ISL) encapsulation, IEEE 802.10 encapsulation, IEEE 802.1Q encapsulation, and multilayer switching (MLS).
Cisco IOS Mobile Wireless Gateway GPRS Support Node Configuration Guide *Cisco IOS Mobile Wireless Gateway GPRS Support Node Command Reference*	Cisco IOS Gateway GPRS Support Node (GGSN) in a 2.5-generation general packet radio service (GPRS) and 3-generation universal mobile telecommunication system (UMTS) network.
Cisco IOS Mobile Wireless Home Agent Configuration Guide *Cisco IOS Mobile Wireless Home Agent Command Reference*	Cisco Mobile Wireless Home Agent, an anchor point for mobile terminals for which mobile IP or proxy mobile IP services are provided.
Cisco IOS Mobile Wireless Packet Data Serving Node Configuration Guide *Cisco IOS Mobile Wireless Packet Data Serving Node Command Reference*	Cisco Packet Data Serving Node (PDSN), a wireless gateway that is between the mobile infrastructure and standard IP networks and that enables packet data services in a code division multiple access (CDMA) environment.
Cisco IOS Mobile Wireless Radio Access Networking Configuration Guide *Cisco IOS Mobile Wireless Radio Access Networking Command Reference*	Cisco IOS radio access network products.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Multiprotocol Label Switching Configuration Guide *Cisco IOS XE Multiprotocol Label Switching Configuration Guide* *Cisco IOS Multiprotocol Label Switching Command Reference*	MPLS Label Distribution Protocol (LDP), MPLS Layer 2 VPNs, MPLS Layer 3 VPNs, MPLS Traffic Engineering (TE), and MPLS Embedded Management (EM) and MIBs.
Cisco IOS Multi-Topology Routing Configuration Guide *Cisco IOS Multi-Topology Routing Command Reference*	Unicast and multicast topology configurations, traffic classification, routing protocol support, and network management support.
Cisco IOS NetFlow Configuration Guide *Cisco IOS XE NetFlow Configuration Guide* *Cisco IOS NetFlow Command Reference*	Network traffic data analysis, aggregation caches, export features.
Cisco IOS Network Management Configuration Guide *Cisco IOS XE Network Management Configuration Guide* *Cisco IOS Network Management Command Reference*	Basic system management; system monitoring and logging; troubleshooting, logging, and fault management; Cisco Discovery Protocol; Cisco IOS Scripting with Tool Control Language (Tcl); Cisco networking services (CNS); DistributedDirector; Embedded Event Manager (EEM); Embedded Resource Manager (ERM); Embedded Syslog Manager (ESM); HTTP; Remote Monitoring (RMON); SNMP; and VPN Device Manager Client for Cisco IOS Software (XSM Configuration).
Cisco IOS Novell IPX Configuration Guide *Cisco IOS XE Novell IPX Configuration Guide* *Cisco IOS Novell IPX Command Reference*	Novell Internetwork Packet Exchange (IPX) protocol.
Cisco IOS Optimized Edge Routing Configuration Guide *Cisco IOS Optimized Edge Routing Command Reference*	Optimized edge routing (OER) monitoring, policy configuration, routing control, logging and reporting, and VPN IPsec/generic routing encapsulation (GRE) tunnel interface optimization.
Cisco IOS Quality of Service Solutions Configuration Guide *Cisco IOS XE Quality of Service Solutions Configuration Guide* *Cisco IOS Quality of Service Solutions Command Reference*	Class-based weighted fair queuing (CBWFQ), custom queuing, distributed traffic shaping (DTS), generic traffic shaping (GTS), IP- to-ATM class of service (CoS), low latency queuing (LLQ), modular QoS CLI (MQC), Network-Based Application Recognition (NBAR), priority queuing, Security Device Manager (SDM), Multilink PPP (MLPPP) for QoS, header compression, AutoQoS, QoS features for voice, Resource Reservation Protocol (RSVP), weighted fair queuing (WFQ), and weighted random early detection (WRED).
Cisco IOS Security Configuration Guide *Cisco IOS XE Security Configuration Guide* *Cisco IOS Security Command Reference*	Access control lists (ACLs), authentication, authorization, and accounting (AAA), firewalls, IP security and encryption, neighbor router authentication, network access security, network data encryption with router authentication, public key infrastructure (PKI), RADIUS, TACACS+, terminal access security, and traffic filters.

Table 1 Cisco IOS and Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
Cisco IOS Service Selection Gateway Configuration Guide *Cisco IOS Service Selection Gateway Command Reference*	Subscriber authentication, service access, and accounting.
Cisco IOS Software Activation Configuration Guide *Cisco IOS Software Activation Command Reference*	An orchestrated collection of processes and components to activate Cisco IOS software feature sets by obtaining and validating Cisco software licenses.
Cisco IOS Software Modularity Installation and Configuration Guide *Cisco IOS Software Modularity Command Reference*	Installation and basic configuration of software modularity images, including installations on single and dual route processors, installation rollbacks, software modularity binding, software modularity processes and patches.
Cisco IOS Terminal Services Configuration Guide *Cisco IOS Terminal Services Command Reference* *Cisco IOS XE Terminal Services Command Reference*	DEC, local-area transport (LAT), and X.25 packet assembler/disassembler (PAD).
Cisco IOS Virtual Switch Command Reference	Virtual switch redundancy, high availability, and packet handling; converting between standalone and virtual switch modes; virtual switch link (VSL); Virtual Switch Link Protocol (VSLP). **Note** For information about virtual switch configuration, refer to the product-specific software configuration information for the Cisco Catalyst 6500 series switch or for the Metro Ethernet 6500 series switch.
Cisco IOS Voice Configuration Library *Cisco IOS Voice Command Reference*	Cisco IOS support for voice call control protocols, interoperability, physical and virtual interface management, and troubleshooting. The library includes documentation for IP telephony applications.
Cisco IOS VPDN Configuration Guide *Cisco IOS XE VPDN Configuration Guide* *Cisco IOS VPDN Command Reference*	Layer 2 Tunneling Protocol (L2TP) dial-out load balancing and redundancy, L2TP extended failover, L2TP security VPDN, multihop by Dialed Number Identification Service (DNIS), timer and retry enhancements for L2TP and Layer 2 Forwarding (L2F), RADIUS Attribute 82: tunnel assignment ID, shell-based authentication of VPDN users, tunnel authentication via RADIUS on tunnel terminator.
Cisco IOS Wide-Area Networking Configuration Guide *Cisco IOS XE Wide-Area Networking Configuration Guide* *Cisco IOS Wide-Area Networking Command Reference*	Frame Relay, Layer 2 Tunneling Protocol Version 3 (L2TPv3), Link Access Procedure, Balanced (LAPB), Switched Multimegabit Data Service (SMDS), and X.25.
Cisco IOS Wireless LAN Configuration Guide *Cisco IOS Wireless LAN Command Reference*	Broadcast key rotation, IEEE 802.11x support, IEEE 802.1x authenticator, IEEE 802.1x local authentication service for Extensible Authentication Protocol-Flexible Authentication via Secure Tunneling (EAP-FAST), Multiple Basic Service Set ID (BSSID), Wi-Fi Multimedia (WMM) required elements, and Wi-Fi Protected Access (WPA).

Table 2 Cisco IOS Supplementary Documents and Resources

Document Title	Description
Cisco IOS Master Command List, All Releases	Alphabetical list of all the commands documented in all Cisco IOS releases.
Cisco IOS New, Modified, Removed, and Replaced Commands	List of all the new, modified, removed, and replaced commands for a Cisco IOS release.
Cisco IOS Software System Messages	List of Cisco IOS system messages and descriptions. System messages may indicate problems with your system; be informational only; or may help diagnose problems with communications lines, internal hardware, or the system software.
Cisco IOS Debug Command Reference	Alphabetical list of **debug** commands including brief descriptions of use, command syntax, and usage guidelines.
Release Notes and Caveats	Information about new and changed features, system requirements, and other useful information about specific software releases; information about defects in specific Cisco IOS software releases.
MIBs	Files used for network monitoring. To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator at the following URL: http://www.cisco.com/go/mibs
RFCs	Standards documents maintained by the Internet Engineering Task Force (IETF) that Cisco IOS documentation references where applicable. The full text of referenced RFCs may be obtained at the following URL: http://www.rfc-editor.org/

Additional Resources and Documentation Feedback

What's New in Cisco Product Documentation is published monthly and describes all new and revised Cisco technical documentation. The *What's New in Cisco Product Documentation* publication also provides information about obtaining the following resources:

- Technical documentation
- Cisco product security overview
- Product alerts and field notices
- Technical assistance

Cisco IOS technical documentation includes embedded feedback forms where you can rate documents and provide suggestions for improvement. Your feedback helps us improve our documentation.

Using the Command-Line Interface in Cisco IOS and Cisco IOS XE Software

Last updated: August 6, 2008

This document provides basic information about the command-line interface (CLI) in Cisco IOS and Cisco IOS XE software and how you can use some of the CLI features. This document contains the following sections:

For more information about using the CLI, see the "Using the Cisco IOS Command-Line Interface" section of the *Cisco IOS Configuration Fundamentals Configuration Guide*.

For information about the software documentation set, see the "About Cisco IOS and Cisco IOS XE Software Documentation" document.

Initially Configuring a Device

Initially configuring a device varies by platform. For information about performing an initial configuration, see the hardware installation documentation that is provided with the original packaging of the product or go to the Product Support area of Cisco.com at http://www.cisco.com/web/psa/products/index.html.

After you have performed the initial configuration and connected the device to your network, you can configure the device by using the console port or a remote access method, such as Telnet or Secure Shell (SSH), to access the CLI or by using the configuration method provided on the device, such as Security Device Manager.

Changing the Default Settings for a Console or AUX Port

There are only two changes that you can make to a console port and an AUX port:

- Change the port speed with the **config-register 0x** command. Changing the port speed is not recommended. The well-known default speed is 9600.
- Change the behavior of the port; for example, by adding a password or changing the timeout value.

Note The AUX port on the Route Processor (RP) installed in a Cisco ASR1000 series router does not serve any useful customer purpose and should be accessed only under the advisement of a customer support representative.

Using the CLI

This section describes the following topics:

Understanding Command Modes

The CLI command mode structure is hierarchical, and each mode supports a set of specific commands. This section describes the most common of the many modes that exist.

Table 1 lists common command modes with associated CLI prompts, access and exit methods, and a brief description of how each mode is used.

Table 1 CLI Command Modes

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
User EXEC	Log in.	`Router>`	Issue the **logout** or **exit** command.	• Change terminal settings. • Perform basic tests. • Display device status.
Privileged EXEC	From user EXEC mode, issue the **enable** command.	`Router#`	Issue the **disable** command or the **exit** command to return to user EXEC mode.	• Issue **show** and **debug** commands. • Copy images to the device. • Reload the device. • Manage device configuration files. • Manage device file systems.
Global configuration	From privileged EXEC mode, issue the **configure terminal** command.	`Router(config)#`	Issue the **exit** command or the **end** command to return to privileged EXEC mode.	Configure the device.
Interface configuration	From global configuration mode, issue the **interface** command.	`Router(config-if)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual interfaces.
Line configuration	From global configuration mode, issue the **line vty** or **line console** command.	`Router(config-line)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual terminal lines.

Table 1 CLI Command Modes (continued)

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
ROM monitor	From privileged EXEC mode, issue the **reload** command. Press the **Break** key during the first 60 seconds while the system is booting.	`rommon # >` The # symbol represents the line number and increments at each prompt.	Issue the **continue** command.	• Run as the default operating mode when a valid image cannot be loaded. • Access the fall-back procedure for loading an image when the device lacks a valid image and cannot be booted. • Perform password recovery when a CTRL-Break sequence is issued within 60 seconds of a power-on or reload event.
Diagnostic (available only on the Cisco ASR1000 series router)	The router boots or enters diagnostic mode in the following scenarios. When a Cisco IOS process or processes fail, in most scenarios the router will reload. • A user-configured access policy was configured using the **transport-map** command, which directed the user into diagnostic mode. • The router was accessed using an RP auxiliary port. • A break signal (**Ctrl-C**, **Ctrl-Shift-6**, or the **send break** command) was entered, and the router was configured to enter diagnostic mode when the break signal was received.	`Router(diag)#`	If a Cisco IOS process failure is the reason for entering diagnostic mode, the failure must be resolved and the router must be rebooted to exit diagnostic mode. If the router is in diagnostic mode because of a transport-map configuration, access the router through another port or using a method that is configured to connect to the Cisco IOS CLI. If the RP auxiliary port was used to access the router, use another port for access. Accessing the router through the auxiliary port is not useful for customer purposes.	• Inspect various states on the router, including the Cisco IOS state. • Replace or roll back the configuration. • Provide methods of restarting the Cisco IOS software or other processes. • Reboot hardware, such as the entire router, an RP, an ESP, a SIP, a SPA, or possibly other hardware components. • Transfer files into or off of the router using remote access methods such as FTP, TFTP, and SCP.

EXEC commands are not saved when the software reboots. Commands that you issue in a configuration mode can be saved to the startup configuration. If you save the running configuration to the startup configuration, these commands will execute when the software is rebooted. Global configuration mode is the highest level of configuration mode. From global configuration mode, you can enter a variety of other configuration modes, including protocol-specific modes.

ROM monitor mode is a separate mode that is used when the software cannot load properly. If a valid software image is not found when the software boots or if the configuration file is corrupted at startup, the software might enter ROM monitor mode. Use the question symbol (?) to view the commands that you can use while the device is in ROM monitor mode.

```
rommon 1 > ?
alias               set and display aliases command
boot                boot up an external process
confreg             configuration register utility
cont                continue executing a downloaded image
context             display the context of a loaded image
cookie              display contents of cookie PROM in hex
.
.
.
rommon 2 >
```

The following example shows how the command prompt changes to indicate a different command mode:

```
Router> enable
Router# configure terminal
Router(config)# interface ethernet 1/1
Router(config-if)# ethernet
Router(config-line)# exit
Router(config)# end
Router#
```

Note A keyboard alternative to the **end** command is Ctrl-Z.

Using the Interactive Help Feature

The CLI includes an interactive Help feature. Table 2 describes how to use the Help feature.

Table 2 CLI Interactive Help Commands

Command	Purpose
help	Provides a brief description of the help feature in any command mode.
?	Lists all commands available for a particular command mode.
*partial command***?**	Provides a list of commands that begin with the character string (no space between the command and the question mark).
partial command<**Tab**>	Completes a partial command name (no space between the command and <Tab>).
command **?**	Lists the keywords, arguments, or both associated with the command (space between the command and the question mark).
command keyword **?**	Lists the arguments that are associated with the keyword (space between the keyword and the question mark).

The following examples show how to use the help commands:

help

```
Router> help

Help may be requested at any point in a command by entering a question mark '?'.  If
nothing matches, the help list will be empty and you must backup until entering a '?'
shows the available options.

Two styles of help are provided:

1. Full help is available when you are ready to enter a command argument (e.g. 'show ?')
and describes each possible argument.

2. Partial help is provided when an abbreviated argument is entered and you want to know
what arguments match the input (e.g. 'show pr?'.)
```

?

```
Router# ?
Exec commands:
  access-enable        Create a temporary access-List entry
  access-profile       Apply user-profile to interface
  access-template      Create a temporary access-List entry
  alps                 ALPS exec commands
  archive              manage archive files
<snip>
```

***partial command*?**

```
Router(config)# zo?
zone  zone-pair
```

***partial command*<Tab>**

```
Router(config)# we<Tab> webvpn
```

***command* ?**

```
Router(config-if)# pppoe ?
  enable        Enable pppoe
  max-sessions  Maximum PPPOE sessions
```

***command keyword* ?**

```
Router(config-if)# pppoe enable ?
  group  attach a BBA group
  <cr>
```

Understanding Command Syntax

Command syntax is the format in which a command should be entered in the CLI. Commands include the name of the command, keywords, and arguments. Keywords are alphanumeric strings that are used literally. Arguments are placeholders for values that a user must supply. Keywords and arguments may be required or optional.

Specific conventions convey information about syntax and command elements. Table 3 describes these conventions.

Table 3 CLI Syntax Conventions

Symbol/Text	Function	Notes
< > (angle brackets)	Indicate that the option is an argument.	Sometimes arguments are displayed without angle brackets.
A.B.C.D.	Indicates that you must enter a dotted decimal IP address.	Angle brackets (< >) are not always used to indicate that an IP address is an argument.
WORD (all capital letters)	Indicates that you must enter one word.	Angle brackets (< >) are not always used to indicate that a WORD is an argument.
LINE (all capital letters)	Indicates that you must enter more than one word.	Angle brackets (< >) are not always used to indicate that a LINE is an argument.
<cr> (carriage return)	Indicates the end of the list of available keywords and arguments, and also indicates when keywords and arguments are optional. When <cr> is the only option, you have reached the end of the branch or the end of the command if the command has only one branch.	—

The following examples show syntax conventions:

```
Router(config)# ethernet cfm domain ?
  WORD  domain name
Router(config)# ethernet cfm domain dname ?
  level
Router(config)# ethernet cfm domain dname level ?
  <0-7>  maintenance level number
Router(config)# ethernet cfm domain dname level 7 ?
  <cr>
Router(config)# snmp-server file-transfer access-group 10 ?
  protocol  protocol options
  <cr>
Router(config)# logging host ?
  Hostname or A.B.C.D  IP address of the syslog server
  ipv6                 Configure IPv6 syslog server
Router(config)# snmp-server file-transfer access-group 10 ?
  protocol  protocol options
  <cr>
```

Understanding Enable and Enable Secret Passwords

Some privileged EXEC commands are used for actions that impact the system, and it is recommended that you set a password for these commands to prevent unauthorized use. Two types of passwords, enable (not encrypted) and enable secret (encrypted), can be set. The following commands set these passwords and are issued in global configuration mode:

- **enable** *password*
- **enable secret** *password*

Using an enable secret password is recommended because it is encrypted and more secure than the enable password. When you use an enable secret password, text is encrypted (unreadable) before it is written to the config.text file. When you use an enable password, the text is written as entered (readable) to the config.text file.

Each type of password is case sensitive, can contain from 1 to 25 uppercase and lowercase alphanumeric characters, and can start with a number. Spaces are also valid password characters; for example, "two words" is a valid password. Leading spaces are ignored, but trailing spaces are recognized.

Note Both password commands have numeric keywords that are single integer values. If you choose a number for the first character of your password followed by a space, the system will read the number as if it were the numeric keyword and not as part of your password.

When both passwords are set, the enable secret password takes precedence over the enable password.

To remove a password, use the **no** form of the commands: **no enable** *password* or **no enable secret** *password*.

For more information about password recovery procedures for Cisco products, see http://www.cisco.com/en/US/products/sw/iosswrel/ps1831/products_tech_note09186a00801746e6.shtml.

Using the Command History Feature

The CLI command history feature saves the commands you enter during a session in a command history buffer. The default number of commands saved is 10, but the number is configurable within the range of 0 to 256. This command history feature is particularly useful for recalling long or complex commands.

To change the number of commands saved in the history buffer for a terminal session, issue the **terminal history size** command:

```
Router# terminal history size num
```

A command history buffer is also available in line configuration mode with the same default and configuration options. To set the command history buffer size for a terminal session in line configuration mode, issue the **history** command:

```
Router(config-line)# history [size num]
```

To recall commands from the history buffer, use the following methods:

- Press Ctrl-P or the up arrow key—Recalls commands beginning with the most recent command. Repeat the key sequence to recall successively older commands.

- Press Ctrl-N or the down arrow key—Recalls the most recent commands in the history buffer after they have been recalled using Ctrl-P or the up arrow key. Repeat the key sequence to recall successively more recent commands.

Note The arrow keys function only on ANSI-compatible terminals such as the VT100.

- Issue the **show history** command in user EXEC or privileged EXEC mode—Lists the most recent commands that you entered. The number of commands that are displayed is determined by the setting of the **terminal history size** and **history** commands.

  The CLI command history feature is enabled by default. To disable this feature for a terminal session, issue the **terminal no history** command in user EXEC or privileged EXEC mode or the **no history** command in line configuration mode.

Abbreviating Commands

Typing a complete command name is not always required for the command to execute. The CLI recognizes an abbreviated command when the abbreviation contains enough characters to uniquely identify the command. For example, the **show version** command can be abbreviated as **sh ver**. It cannot be abbreviated as **s ver** because **s** could mean **show**, **set**, or **systat**. The **sh v** abbreviation also is not valid because the **show** command has **vrrp** as a keyword in addition to **version**. (Command and keyword examples from Cisco IOS Release 12.4(13)T.)

Using Aliases for CLI Commands

To save time and the repetition of entering the same command multiple times, you can use a command alias. An alias can be configured to do anything that can be done at the command line, but an alias cannot move between modes, type in passwords, or perform any interactive functions.

Table 4 shows the default command aliases.

Table 4 Default Command Aliases

Command Alias	Original Command
h	help
lo	logout
p	ping
s	show
u or **un**	undebug
w	where

To create a command alias, issue the **alias** command in global configuration mode. The syntax of the command is **alias** *mode command-alias original-command*. Following are some examples:

- Router(config)# **alias exec prt partition**—privileged EXEC mode
- Router(config)# **alias configure sb source-bridge**—global configuration mode
- Router(config)# **alias interface rl rate-limit**—interface configuration mode

To view both default and user-created aliases, issue the **show alias** command.

For more information about the **alias** command, see http://www.cisco.com/en/US/docs/ios/fundamentals/command/reference/cf_book.html.

Using the no and default Forms of Commands

Most configuration commands have a **no** form that is used to reset a command to its default value or disable a feature or function. For example, the **ip routing** command is enabled by default. To disable this command, you would issue the **no ip routing** command. To re-enable IP routing, you would issue the **ip routing** command.

Configuration commands may also have a **default** form, which returns the command settings to their default values. For commands that are disabled by default, using the **default** form has the same effect as using the **no** form of the command. For commands that are enabled by default and have default settings, the **default** form enables the command and returns the settings to their default values.

The **no** and **default** forms of commands are described in the command pages of command references.

Using the debug Command

A **debug** command produces extensive output that helps you troubleshoot problems in your network. These commands are available for many features and functions within Cisco IOS and Cisco IOS XE software. Some **debug** commands are **debug all**, **debug aaa accounting**, and **debug mpls packets**. To use **debug** commands during a Telnet session with a device, you must first enter the **terminal monitor** command. To turn off debugging completely, you must enter the **undebug all** command.

For more information about **debug** commands, see the *Cisco IOS Debug Command Reference* at http://www.cisco.com/en/US/docs/ios/debug/command/reference/db_book.html.

Caution Debugging is a high priority and high CPU utilization process that can render your device unusable. Use **debug** commands only to troubleshoot specific problems. The best times to run debugging are during periods of low network traffic and when few users are interacting with the network. Debugging during these periods decreases the likelihood that the **debug** command processing overhead will affect network performance or user access or response times.

Filtering Output Using Output Modifiers

Many commands produce lengthy output that may use several screens to display. Using output modifiers, you can filter this output to show only the information that you want to see.

Three output modifiers are available and are described as follows:

- **begin** *regular expression*—Displays the first line in which a match of the regular expression is found and all lines that follow.
- **include** *regular expression*—Displays all lines in which a match of the regular expression is found.
- **exclude** *regular expression*—Displays all lines except those in which a match of the regular expression is found.

To use one of these output modifiers, type the command followed by the pipe symbol (|), the modifier, and the regular expression that you want to search for or filter. A regular expression is a case-sensitive alphanumeric pattern. It can be a single character or number, a phrase, or a more complex string.

The following example illustrates how to filter output of the **show interface** command to display only lines that include the expression "protocol."

```
Router# show interface | include protocol

FastEthernet0/0 is up, line protocol is up
Serial4/0 is up, line protocol is up
Serial4/1 is up, line protocol is up
Serial4/2 is administratively down, line protocol is down
Serial4/3 is administratively down, line protocol is down
```

Understanding CLI Error Messages

You may encounter some error messages while using the CLI. Table 5 shows the common CLI error messages.

Table 5 Common CLI Error Messages

Error Message	Meaning	How to Get Help
% Ambiguous command: "show con"	You did not enter enough characters for the command to be recognized.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Incomplete command.	You did not enter all the keywords or values required by the command.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Invalid input detected at "^" marker.	You entered the command incorrectly. The caret (^) marks the point of the error.	Enter a question mark (?) to display all the commands that are available in this command mode. The keywords that you are allowed to enter for the command appear.

For more system error messages, see the following documents:

- *Cisco IOS Release 12.2SR System Message Guide*
- *Cisco IOS System Messages, Volume 1 of 2* (Cisco IOS Release 12.4)
- *Cisco IOS System Messages, Volume 2 of 2* (Cisco IOS Release 12.4)

Saving Changes to a Configuration

To save changes that you made to the configuration of a device, you must issue the **copy running-config startup-config** command or the **copy system:running-config nvram:startup-config** command. When you issue these commands, the configuration changes that you made are saved to the startup configuration and saved when the software reloads or power to the device is turned off or interrupted. The following example shows the syntax of the **copy running-config startup-config** command:

```
Router# copy running-config startup-config
Destination filename [startup-config]?
```

You press Enter to accept the startup-config filename (the default), or type a new filename and then press Enter to accept that name. The following output is displayed indicating that the configuration was saved:

```
Building configuration...
[OK]
Router#
```

On most platforms, the configuration is saved to NVRAM. On platforms with a Class A flash file system, the configuration is saved to the location specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM.

Additional Information

- "Using the Cisco IOS Command-Line Interface" section of the *Cisco IOS Configuration Fundamentals Configuration Guide*:

  http://www.cisco.com/en/US/docs/ios/fundamentals/configuration/guide/cf_cli-basics.html

  or

  "Using Cisco IOS XE Software" chapter of the *Cisco ASR1000 Series Aggregation Services Routers Software Configuration Guide*:

  http://www.cisco.com/en/US/docs/routers/asr1000/configuration/guide/chassis/using_cli.html
- Cisco Product Support Resources

  http://www.cisco.com/web/psa/products/index.html
- Support area on Cisco.com (also search for documentation by task or product)

  http://www.cisco.com/en/US/support/index.html
- *White Paper: Cisco IOS Reference Guide*

  http://www.cisco.com/en/US/products/sw/iosswrel/ps1828/products_white_paper09186a008018305e.shtml
- Software Download Center (downloads; tools; licensing, registration, advisory, and general information) (requires Cisco.com User ID and password)

  http://www.cisco.com/kobayashi/sw-center/
- Error Message Decoder, a tool to help you research and resolve error messages for Cisco IOS software

  http://www.cisco.com/pcgi-bin/Support/Errordecoder/index.cgi

- Command Lookup Tool, a tool to help you find detailed descriptions of Cisco IOS commands (requires Cisco.com user ID and password)

  http://tools.cisco.com/Support/CLILookup

- Output Interpreter, a troubleshooting tool that analyzes command output of supported **show** commands

  https://www.cisco.com/pcgi-bin/Support/OutputInterpreter/home.pl\

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.

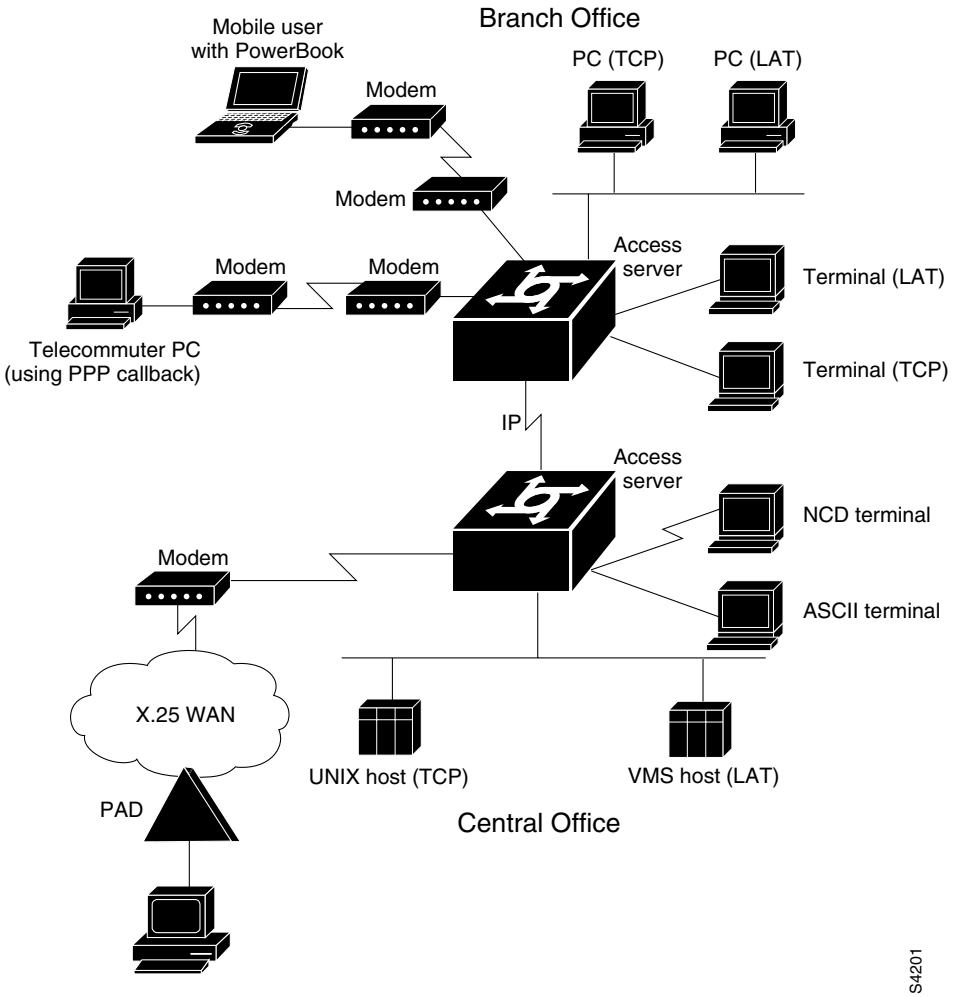
Figure 1 illustrates the following access services available in the Cisco IOS software:

- 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).

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.

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

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.

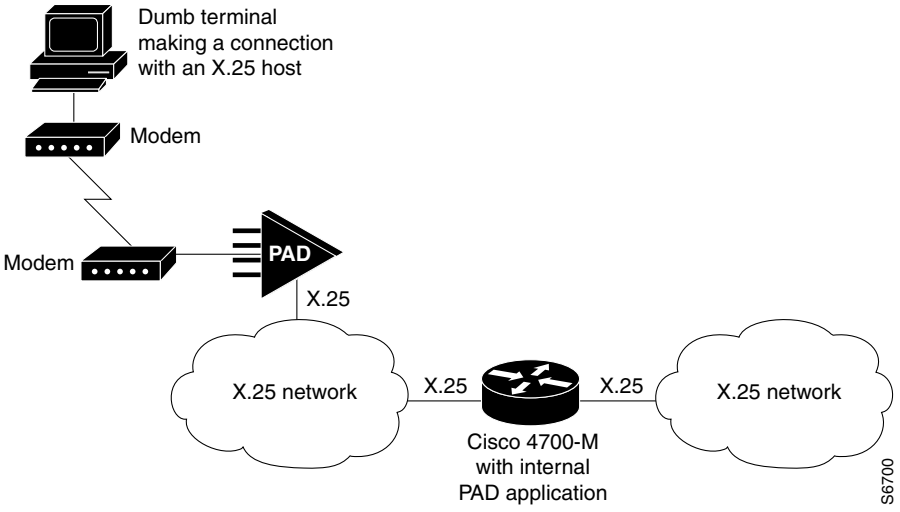
The chapters in this publication describe how to configure the lines for a specific device application. See the chapter "Configuring Protocol Translation and Virtual Asynchronous Devices" in this publication, and the chapters "Interfaces, Controllers, and Lines Used for Dial Access Overview" and "Preparing 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



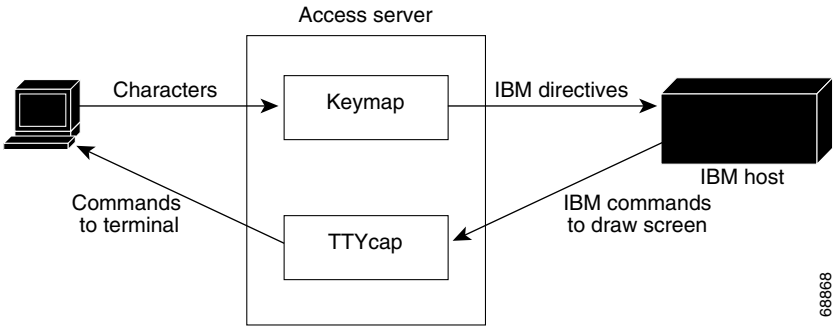
Remote Node Services

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

- AppleTalk Remote Access (ARA), which is described in the chapter “Configuring AppleTalk Remote Access” in this publication.

  Using ARA, Macintosh users can connect across telephone lines into an AppleTalk network to access network resources, such as printers, file servers, and e-mail. Remote users running ARA have the same access to network resources as a Macintosh connected directly to the LAN. They can also run other applications on top of ARA to access UNIX file servers for such tasks as reading e-mail and copying or transferring files between UNIX hosts. Note that Macintosh users can run Macintosh-based SLIP or PPP applications to access non-AppleTalk-based resources (see Figure 3).

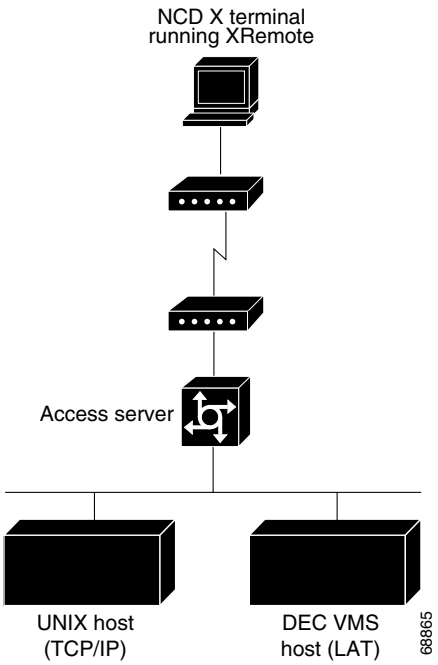
Figure 3 Remote Node Connection—Macintosh and PC Users Dialing In



- 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 4 illustrates an XRemote connection.

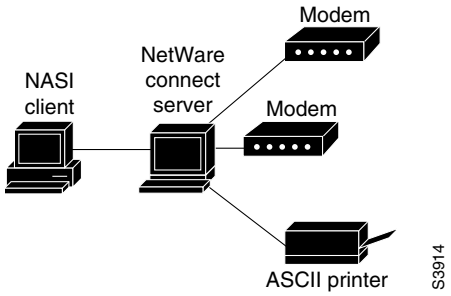
Figure 4 XRemote Connection



- 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 5.)

Figure 5 NASI Setup in a NetWare Environment



Terminal Services

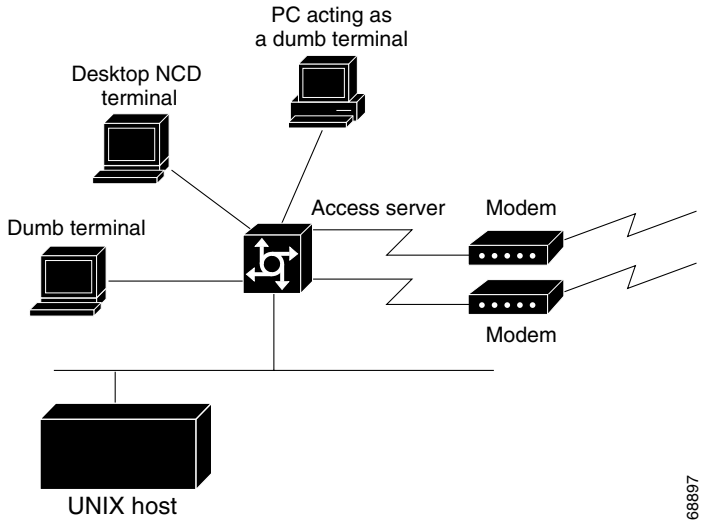
Terminal services permit asynchronous devices to be connected to a LAN or WAN through network and terminal-emulation software including Telnet, rlogin, NASI, the Digital local-area transport (LAT) protocol, and IBM TN3270. (See Figure 6.)

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 6 Terminal-to-Host Connectivity



Protocol Translation

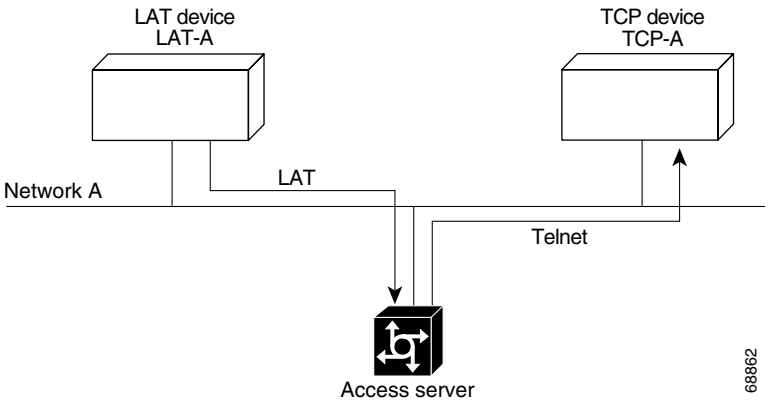
Protocol translation services are essentially an extension of terminal services. A user running a TCP/IP-based application can connect to a host running a different virtual terminal protocol, such as the Digital LAT protocol. The Cisco IOS software converts one virtual terminal protocol into another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

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 7 illustrates LAT-to-TCP protocol translation.

Figure 7 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.

The Cisco PAD Facility for X.25 Connections

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

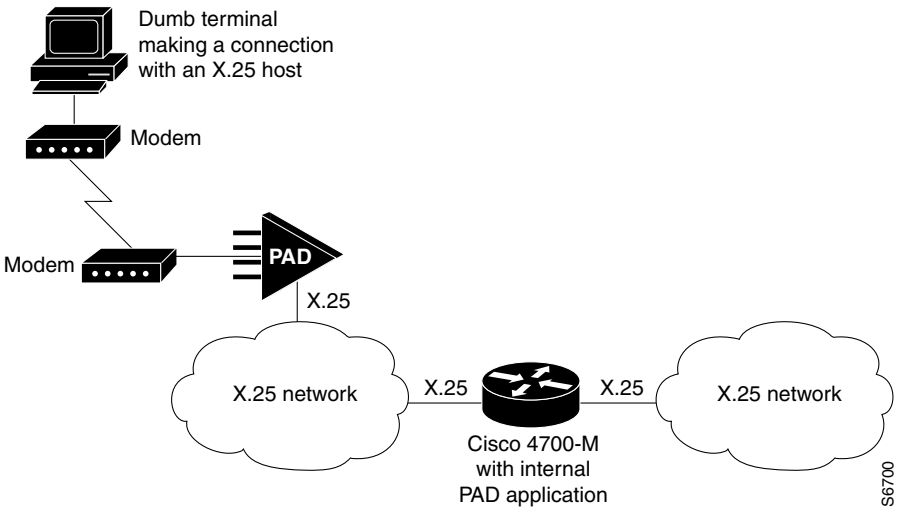
PADs are configured to enable X.25 connections between network devices. A PAD is a device that receives a character stream from one or more terminals, assembles the character stream into packets, and sends the data packets out to a host. A PAD can also do the reverse. It can take data packets from a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

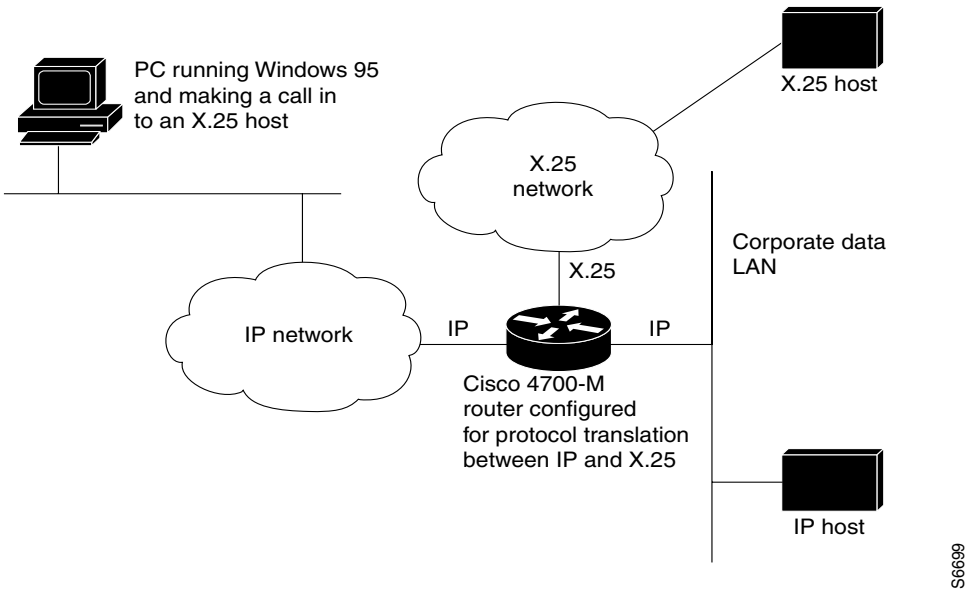
Figure 1 shows a remote X.25 user placing a call through an X.25 switched network to the internal PAD application on a Cisco 4700-M router, and to an X.25 host located inside a corporate data center.

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.

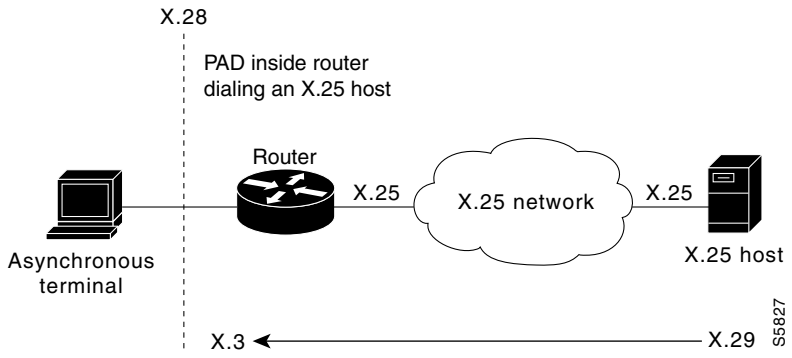
Cisco Universal X.28 PAD Emulation Mode

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.

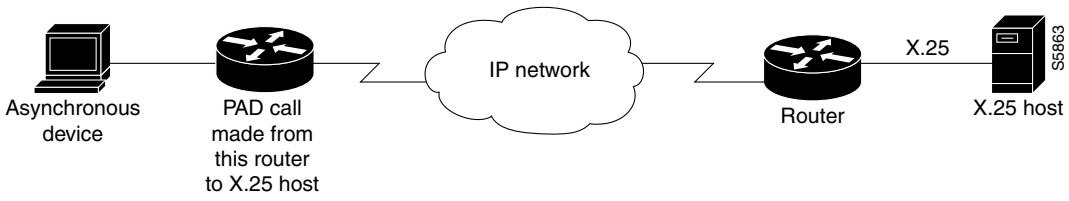
```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# line vty 0 4
Router(config-line)# autocommand x28
Router(config-line)# rotary 1
```

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
```

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
12.3(2)T	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 the "Configuring the Cisco PAD Facility for X.25 Connections" chapter in the *Cisco IOS Terminal Services Configuration Guide*.

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 PAD subaddressing	"Configuring PAD Subaddressing" chapter in the *Cisco IOS Terminal Services Configuration Guide*
Information on configuring PAD	"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

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**

Support for NASI Clients to Access Network Resources

Configuring Support for NASI Clients to Access Network Resources

This chapter describes how to allow your router to function as a NetWare Asynchronous Support Interface (NASI) server. It includes the following main sections:

- NASI Server Overview
- Configuring the Router as a NASI Server

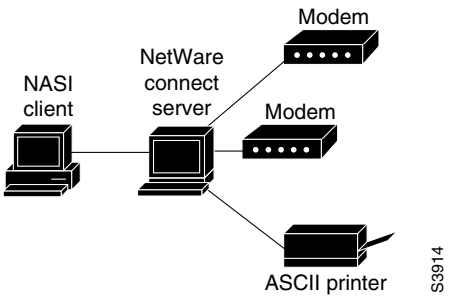
For a complete description of the commands mentioned in this chapter, refer to the *Cisco IOS Terminal Services Command Reference*, Release 12.2. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

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.

NASI Server Overview

A NASI server enables a NASI client to connect to asynchronous network resources (such as modems) without the need for these resources to be located on the desktop of the client. (See Figure 1.)

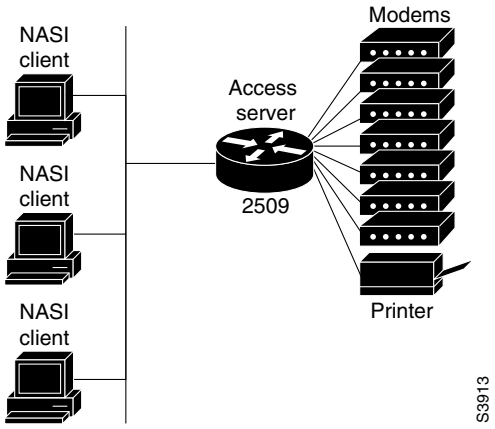
Figure 1 NASI Setup in a NetWare Environment



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

You can configure the Cisco IOS software to enable NASI clients to connect to asynchronous resources attached to your router. The NASI client can connect to any port on the router other than the console port to access network resources (see Figure 2). The 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, beginning with tty1. The user selects the desired outgoing terminal and virtual terminal port. TACACS+ security also can be configured on the router so that after the user selects a terminal and virtual terminal port, a username and password prompt appear for authentication, authorization, and accounting (AAA).

Figure 2 NASI Clients Accessing Asynchronous Resources Through an Access Server



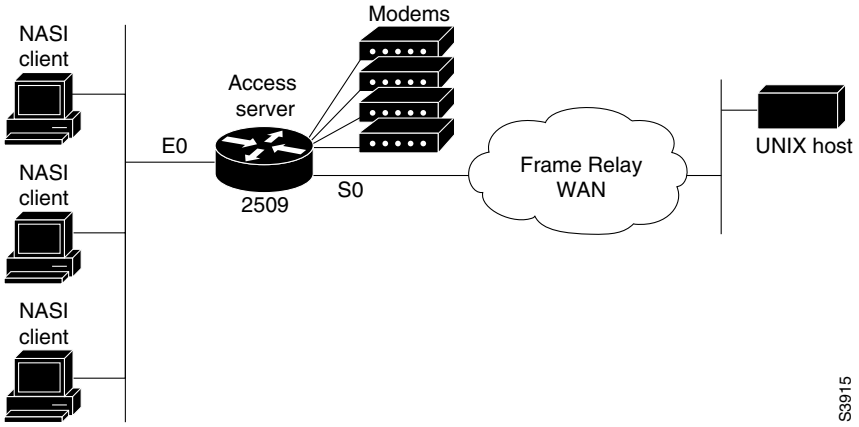
Note The Cisco IOS implementation of NASI functions best with NASI client software version 2.0 and later versions.

The NASI client can be on a local LAN or can be on a remote LAN. If it is on a remote LAN, the following two requirements must be met:

- A router routing Internet Protocol Exchange (IPX) forwards NetWare Connect Server Service Advertising Protocol (SAP) advertisements from the remote LAN to the LAN to which the local router is connected.
- The same router routing IPX spoofs Get Nearest Server (GNS) replies for the GNS requests that the client sends out.

The fact that you can connect to many different ports on the router means that you can provide access to more than one asynchronous device. When the user accesses the vty, the user can connect to the user EXEC facility and issue a Telnet or NASI command to access a remote network (see Figure 3). Only the first available vty appears in the list of available ports on the router (and it is named RCONSOLE).

Figure 3 NASI Clients Gaining Access to IP Hosts on a Remote Network



Configuring the Router as a NASI Server

To configure your router as a NASI server, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing`	Enables IPX routing on the router.
Step 2	`Router(config)# ipx internal-network`	Defines an internal IPX network number.
Step 3	`Router(config)# interface type number`	Enters interface configuration mode.
Step 4	`Router(config-if)# ipx network [network | unnumbered]`	Enables IPX routing on an interface.
Step 5	`Router(config-if)# exit`	Exits to global configuration mode.
Step 6	`Router(config)# ipx nasi-server enable`	Enables NASI.
Step 7	`Router(config)# aaa authentication nasi {list-name | default} {methods list}`	(Optional) Configures TACACS+ security on all lines on the router.
Step 8	`Router(config)# line [aux | tty | vty] line-number [ending-line-number]`	Enters line configuration mode.
Step 9	`Router(config-line)# login authentication nasi {list-name | default}`	(Optional) Configures TACACS+ security on a per-line basis.

You also can configure SAP filters to filter SAP updates, and access lists to filter NASI traffic between interfaces on the router.

Note If a NASI server is already on the LAN segment connected to the router, the router cannot respond to GNS requests for NASI services.

If you have configured NASI on your router, you can use IPX client applications to make IPX dial-out connections to a shared pool of asynchronous devices. For example, a NASI client on the LAN can connect to a serial (synchronous or asynchronous) port on the router, which provides access to remote modems, printers, and networks. The command the user issues depends on the application being used to connect to the NASI server. NASI relies on Sequenced Packet Exchange (SPX).

Configuring Protocol Translation and Virtual Asynchronous Devices

This chapter describes how to configure protocol translation and virtual asynchronous connections using Cisco IOS software. These tasks are described in the following sections, which also describe the process of tunneling and protocol translation, and the two-step and the one-step translation methods:

The X.3 packet assembler/disassembler (PAD) parameters are described in the “X.3 PAD Parameters” appendix later in this publication.

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.

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

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 Telnet functionality.

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 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.

Protocol Translation Overview

This section describes the additional tasks required to perform protocol translation from one host to another host or to a router. It includes the following sections:

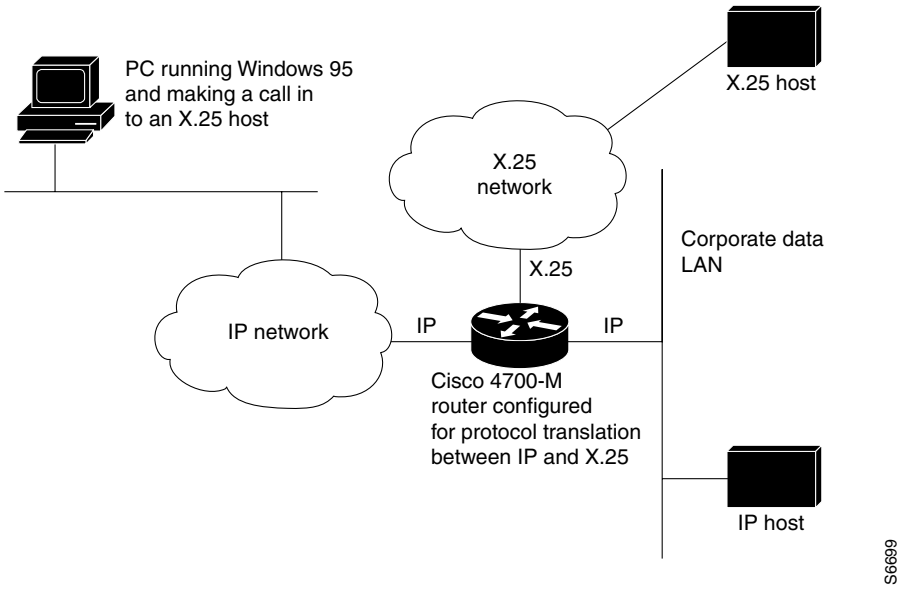
- Definition of Protocol Translation
- Definition of Tunneling
- Deciding Whether to Use One-Step or Two-Step Protocol Translation
- One-Step Protocol Translation
- Two-Step Protocol Translation
- Tunneling SLIP, PPP, and ARA
- Setting Up Virtual Templates for Protocol Translation

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 the 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

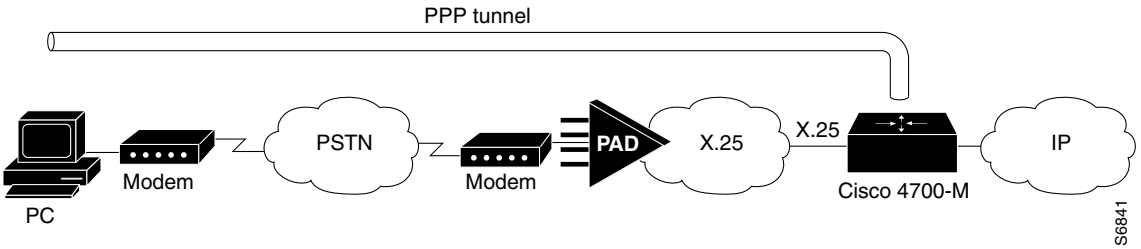


Definition of Tunneling

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

The 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, 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 which host the connection is for and which protocol that 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 the 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 section "Protocol Translation Configuration Task List" later in this chapter 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, we recommend that you configure virtual templates for this feature.

Note You must 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 section "Protocol Translation Configuration Task List" later in this chapter 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 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 section "Configuring Tunneling of SLIP, PPP, or ARA" later in this chapter for configuration tasks.

One-Step Tunneling of SLIP, PPP, and ARA

To use one-step protocol translation to tunnel SLIP, PPP (or IPX-PPP), or ARA, you need not enter any preliminary commands. Simply use the **translate** command with the **slip** or **ppp** keyword for one-step SLIP or PPP connections or the **autocommand arap** command for one-step ARA connections. Because ARA does not use addressing, you must specify the **autocommand** keyword, then specify the string **arap** to tunnel ARA to an AppleTalk network.

If you are tunneling PPP, SLIP, or ARA across X.25, you must also set up your X.3 profile correctly using the **x29 profile** command, as described in the section "Configuring One-Step Tunneling of SLIP or PPP" later in this chapter.

Two-Step Tunneling of PPP and SLIP

To tunnel SLIP or PPP across an X.25 WAN to an IP network using the two-step protocol translation method, use the **vty-async** command, which enables you to run PPP and SLIP on virtual terminal lines. Normally, PPP and SLIP function only on physical asynchronous interfaces. The **vty-async** command enables you to run PPP and SLIP on virtual terminal lines, which permits you to tunnel from an incoming protocol to SLIP or PPP and then to an IP network (or IPX-PPP to an IPX network).

If you make a PAD connection to a router running protocol translation and then issue the **ppp** *definitions* command to connect across an X.25 network, you also must set up your X.3 profile using the **pad** [**/profile** *name*] command.

Two-Step Tunneling of ARA

To tunnel ARA using the two-step method, you configure ARA on one or more virtual terminal lines and then configure automatic protocol startup. When a user connects to the vty and receives an EXEC prompt, ARA starts up automatically on the outgoing vty.

Setting Up Virtual Templates for Protocol Translation

The Cisco IOS software simplifies the process of configuring protocol translation to tunnel PPP or SLIP across X.25, TCP, and LAT networks. It does so by providing virtual interface templates that you can configure independently and apply to any protocol translation session. You can configure virtual interface templates for one-step and two-step protocol translation.

A virtual interface template is an interface that exists just inside the router (it is not a physical interface). You can configure virtual interface templates just as you do regular asynchronous serial interfaces. You then apply these virtual interface templates for one-step and two-step protocol translation (the process is described in detail in the section "Protocol Translation Configuration Task List" in this chapter). When a user dials in through a vty and a tunnel connection is established, the router clones the attributes of the virtual interface template onto a *virtual access interface*. This virtual access interface is a temporary interface that supports the asynchronous protocol configuration specified in the virtual interface template. This virtual access interface is created dynamically and lasts only as long as the tunnel session is active.

Before virtual templates were implemented, you enabled asynchronous protocol functions on virtual terminal lines by creating virtual *asynchronous* interfaces rather than virtual *access* interfaces. (For one-step translation, you did so by specifying **ppp** or **slip** as outgoing options in the **translate** command. For two-step translation, you did so by specifying the **vty-async** command.) The differences between virtual asynchronous interfaces and virtual access interfaces are as follows:

- Virtual asynchronous interfaces are allocated permanently, whereas virtual access interfaces are created dynamically when a user calls in, and are closed down when the connection drops.
- Virtual asynchronous interfaces were unconfigurable and supported only a limited set of protocol translation functions. However, virtual access interfaces are fully configurable via the virtual interface template. All attributes of the virtual interface template are cloned onto the virtual access interface when a call comes in.

Virtual access interfaces replace virtual asynchronous interfaces for both one-step and two-step translation.

You can configure up to 25 virtual interface templates and have up to 300 virtual access interfaces per router (300 is the hardware limit on the router, based on the number of IDBs).

Note You can configure only a single virtual interface template (which applies to all virtual terminal asynchronous lines) when tunneling PPP or SLIP using two-step protocol translation.

Figure 3 shows a typical network diagram for a tunnel session from a PC across an X.25 network, through a router set up with a virtual interface template for protocol translation, and to a corporate intranet.

Figure 3 PPP Tunnel Session Across an X.25 Network

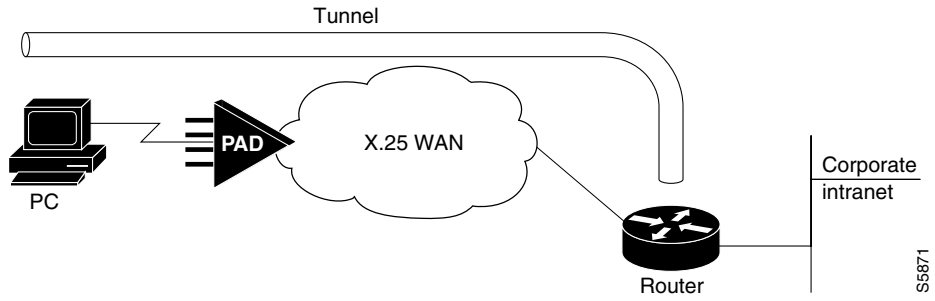
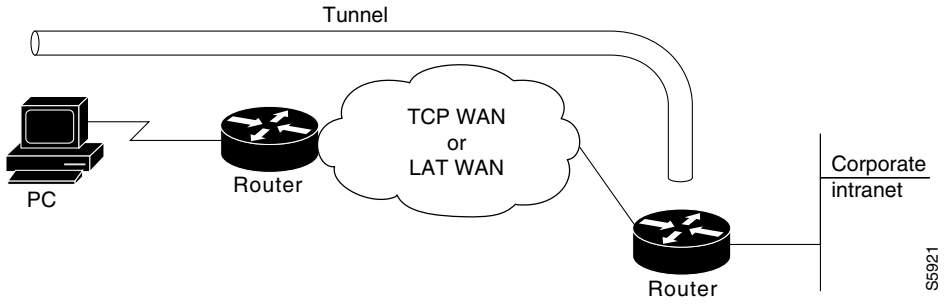


Figure 4 shows a typical network diagram for a tunnel session from a PC across a TCP or LAT WAN, through a router set up with a virtual interface template for protocol translation, and to a corporate intranet.

Figure 4 PPP Tunnel Session Across a TCP or LAT WAN



The virtual interface template service for protocol translation provides the following benefits:

- Allows customized configurations to be predefined in one location, then applied dynamically to any protocol translation session, whether one-step or two-step, for easier maintenance.
- Simplifies the **translate** command syntax by reducing the number of options required within each command.
- Makes virtual asynchronous interfaces configurable for both one-step and two-step protocol translation.

Virtual Templates and L2F

L2F tunneling technology is used in virtual private dialup networks (VPDNs). VPDN allows separate and autonomous protocol domains to share common access infrastructure including modems, access servers, and ISDN routers by the tunneling of link level frames.

L2F/VPDN over protocol translation virtual template interfaces allows services with multiple X.25 dial point of presences (POPs) to expand their current L2F services. This ability can be accomplished by terminating the PPP virtual-asynchronous connections over X.25 at the Cisco protocol translation/router and setting up the L2F tunnel to the home gateway. With this configuration, protocol-level packets are allowed to pass through the virtual tunnel between endpoints of a point-to-point connection.

Typical L2F tunneling use includes Internet service providers (ISPs) or other access service creating virtual tunnels to link to the remote sites of a customer or remote users with corporate home networks. In particular, a network access server at the POP for the ISP exchanges PPP messages with the remote users, and communicates by L2F requests and responses with the home gateway of the customer to set up tunnels.

Frames from the remote users are accepted by the POP, stripped of any linked framing or transparency bytes, encapsulated in L2F, and forwarded over the appropriate tunnel. The home gateway of the customer accepts these L2F frames, strips the L2F encapsulation, and processes the incoming frames for the appropriate interface.

Note This implementation of VPDN supports PPP dialup only.

For more information on VPDNs, refer to the chapters in the part "Virtual Private Networks" in the *Cisco IOS Dial Technologies Configuration Guide*, Release 12.2.

Protocol Translation Configuration Task List

To configure protocol translation, perform the tasks described in the following sections as needed:

- Configuring One-Step Protocol Translation (As Required)
- Configuring a Virtual Template for One-Step Protocol Translation (As Required)
- Configuring Two-Step Protocol Translation (As Required)
- Configuring a Virtual Template for Two-Step Protocol Translation (As Required)

Refer to the sections "Virtual Template for Protocol Translation Examples," "Protocol Translation Application Examples," and "Protocol Translation Session Examples" later in this chapter for examples of protocol translation sessions and configurations.

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 translation.

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 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 One-Step Protocol Translation

To configure a virtual interface template to enable tunneling of PPP or SLIP across an X.25, TCP, or LAT WAN, first create and configure a virtual interface template, then apply it as the single outgoing option to the **translate** command.

Virtual interface templates in general support all commands available on any serial interface, because virtual templates are used for purposes other than protocol translation. However, a virtual access interface—which clones the configuration of the corresponding virtual interface template when created for protocol translation—supports only asynchronous protocol commands.

To enable tunneling of PPP or SLIP across an X.25, TCP, or LAT WAN by using one-step protocol translation, use the following commands beginning 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 {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`	Exits to global configuration mode.
Step 6	`Router(config)# translate {lat | tcp | x25} incoming-address [in-options] virtual-template number [global-options]`	Assigns the virtual interface template to a protocol translation session.

1. You can also assign a specific IP address by using the **ip address** command, though assigning the IP address of the Ethernet 0 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**.

Rather than specify outgoing translation options in the **translate** command, configure these options as interface configuration commands under the virtual interface template, then apply the virtual interface template to the **translate** command. Table 7 maps outgoing **translate** command options to interface commands you can configure in the virtual interface template.

Table 7 Mapping Outgoing translate Command Options to Interface Commands

translate Command Options	Corresponding Interface Configuration Command
ip-pool	**peer default ip address** {**dhcp** | **pool** [*pool-name-list*]}
header-compression	**ip tcp header compression** [**on** | **off** | **passive**]
routing	**ip routing** or **ipx routing**
mtu	**mtu**
keepalive	**keepalive**
authentication {**chap** | **pap**}	**ppp authentication** {**chap** | **pap**}
ppp use-tacacs	**ppp use-tacacs**
ipx loopback	**ipx ppp-client loopback** *number*

Configuring Two-Step Protocol Translation

To translate using the two-step method, use the following commands in EXEC mode. The first step is required only if you are tunneling SLIP or PPP using the two-step protocol translation facility.

	Command	Purpose
Step 1	`Router>` **`connect`** or `Router>` **`lat`** or `Router>` **`pad`** or `Router>` **`telnet`** or `Router>` **`tunnel`**	Establishes an incoming connection to the router running protocol translation.
Step 2	`Router>` **`connect`** or `Router>` **`lat`** or `Router>` **`pad`** or `Router>` **`telnet`** or `Router>` **`tunnel`** or `Router>` **`ppp`** or `Router>` **`slip`**	Establishes the outgoing connection from the router supporting protocol translation to another network host.

The Cisco IOS software supports the two-step method in both directions for protocols other than PPP and SLIP (for example, from Telnet to PAD, and vice versa).

Note PPP and SLIP are supported on outgoing connections only.

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 still use the **vty-async** command, just as before implementation of virtual templates. However, virtual asynchronous interfaces are not created as they were before virtual interface templates. Virtual access 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 beginning 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`	Exits 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. We recommend that you include security on your virtual interface template. For example, you can enter the **ppp authentication chap** 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 up to 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, beginning in global configuration mode:

Command	Purpose
`Router(config)# line vty line-number`	Increases the number of virtual terminal lines, and thus the maximum number of protocol translation sessions.
`Router(config-line)# no line vty line-number`	Decreases the number of virtual terminal lines, and thus the maximum number of protocol translation sessions.

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.

Configuring Tunneling of SLIP, PPP, or ARA

To configure SLIP, PPP, or ARA tunneling, perform the tasks described in the following sections:

- Configuring One-Step Tunneling of SLIP or PPP (As Required)
- Configuring a Virtual Template for One-Step Protocol Translation (As Required)
- Configuring Two-Step Tunneling of SLIP or PPP (As Required)
- Enabling Dynamic Address Assignment for Outgoing PPP and SLIP on Virtual Terminal Lines (As Required)

You can also enable IPX over tunneled PPP sessions.

Configuring One-Step Tunneling of SLIP or PPP

To tunnel SLIP or PPP using the one-step protocol translation facility, use the following commands in global configuration mode:

Command	Purpose
`Router(config)# x29 profile name parameter:value [parameter:value]`	(Optional) If you are tunneling PPP over X.25, creates an X.3 profile so that the router will interoperate with the PAD.
`Router(config)# translate protocol incoming-address [in-options] protocol outgoing-address [out-options] [global-options]`	Creates the connection specifications for one-step protocol translation.

If you are configuring PPP over X.25 and do not know which X.3 profile parameters to use, try the following (these parameters do not function in all cases; they are simply a place from which to start):

1:0, 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, 17:24, 18:18, 19:0, 20:0, 21:0, 22:0

For more information about creating an X.29 profile script, refer to the section "Creating an X.29 Profile Script" later in this chapter. For an example of configuring PPP over X.25, see the section "Tunneling PPP over X.25 Example" at the end of this chapter.

To configure an outgoing session for IPX-PPP, use the **ipx loopback** *number* command for the outgoing session.

To tunnel SLIP or PPP across X.25, LAT, or Telnet using the one-step method, you need not enter any additional commands, as you do when you tunnel SLIP or PPP using the two-step method. The **translate** command enables asynchronous protocol features on one vty at a time.

PPP and SLIP, including IPX-PPP, can be tunneled on outgoing connections only.

Configuring One-Step Tunneling of ARA

To tunnel ARA using the one-step protocol translation facility, use the following commands beginning in global configuration mode. The first four steps are required; steps 5 through 11 are optional:

	Command	Purpose
Step 1	`Router(config)# appletalk routing`	Turns on AppleTalk routing.
Step 2	`Router(config)# translate` *protocol incoming-address* [*in-options*] `autocommand arap`	Uses the protocol translation facility to enable an ARA tunnel across a remote 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 one or more lines.
Step 5	`Router(config-line)# arap dedicated`	Sets one or more dedicated ARA lines.
Step 6	`Router(config-line)# arap timelimit` [*minutes*]	Sets the session time limit.
Step 7	`Router(config-line)# arap warningtime` [*minutes*]	Sets the disconnect warning time.
Step 8	`Router(config-line)# arap noguest`	Disallows guests.
Step 9	`Router(config-line)# arap require-manual-password`	Requires manual password entry.
Step 10	`Router(config-line)# arap zonelist` *zone-access-list-number*	Limits the zones the Macintosh user sees.
Step 11	`Router(config-line)# arap net-access-list` *net-access-list number*	Controls access to networks.

Configuring Two-Step Tunneling of SLIP or PPP

To tunnel SLIP or PPP using the two-step protocol translation facility, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# vty-async`	Enables tunneling of PPP and SLIP using two-step protocol translation.
Step 2	`Router(config)# exit`	Exits from global configuration mode into EXEC mode.

	Command	Purpose
Step 3	`Router>` **`connect`** or `Router>` **`lat`** or `Router>` **`pad`** or `Router>` **`telnet`** or `Router>` **`tunnel`**	Establishes an incoming connection to the router running protocol translation.
Step 4	`Router>` **`connect`** or `Router>` **`slip`** or `Router>` **`ppp`** or `Router>` **`tunnel`**	Establish the outgoing connection from the router supporting protocol translation to another network host.

If you want to configure IPX over your PPP sessions on virtual terminal lines, refer to the chapter "Configuring Asynchronous SLIP and PPP" in the *Cisco IOS Dial Technologies Configuration Guide*, Release 12.2.

Enabling Dynamic Address Assignment for Outgoing PPP and SLIP on Virtual Terminal Lines

You can specify IP addresses dynamically from a Dynamic Host Configuration Protocol (DHCP) proxy client or a local IP address pool on outgoing PPP and SLIP sessions on virtual terminal lines.

Assigning IP Addresses Using DHCP

The DHCP client-proxy feature manages a pool of IP addresses available to PPP or SLIP dial-in clients that need not know an IP address to be able to access a system. This feature allows a finite number of IP addresses to be reused quickly and efficiently by many clients. Additional benefits include the ability to maintain sessions, such as Telnet, even when a modem line fails. When the client is autodialed back into the access server or router, the session can be resumed because the same IP address is reissued to the client by the access server or router.

A DHCP proxy client is a Cisco access server or router configured to arbitrate DHCP calls between a DHCP server and a DHCP client. For more information about DHCP proxy clients, refer to the *Cisco IOS IP Configuration Guide*, Release 12.2.

To assign IP addresses using DHCP, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip address-pool dhcp-proxy-client`	Specifies that the router use the DHCP client-proxy.
Step 2	`Router(config)# translate` *protocol incoming-address* `[`*in-options*`] {slip | ppp} ip-pool`	Specifies DHCP pooling for the SLIP or PPP client on the outgoing session.

The name argument is the name of the DHCP proxy client specified with the **ip address-pool dhcp-proxy-client** command.

Assigning IP Addresses Using Local IP Address Pooling

To make temporary IP addresses available for outgoing PPP and SLIP clients on outgoing sessions, you must first specify that the Cisco IOS software use a local IP address pool on all asynchronous interfaces and create one or more local IP address pools. You then assign local pooling as part of the **translate** command. To assign IP addresses dynamically on a virtual asynchronous connection, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ip address-pool local`	Specifies that the router use a local IP address pool on all asynchronous interfaces.
Step 2	`Router(config)# ip local pool` *name begin-ip-address-range* `[`*end-ip-address-range*`]`	Creates one or more local IP address pools.
Step 3	`Router(config)# translate` *protocol incoming-address* `[`*in-options*`] {slip | ppp ip-pool [scope-name` *name*`]}`	Specifies local pooling for the SLIP or PPP client on the outgoing session.

The **scope-name** option takes the name of any local IP address pool that has been defined using the **ip local pool** command.

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 (Required)
- Applying an Access List to a Virtual Line (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 access and incoming PAD access. 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, you can 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, refer to the section "Tunneling PPP over X.25 Example" later in this chapter.

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, the 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 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 the *Cisco IOS Terminal Services Command Reference*, Release 12.2.

Defining X.25 Host Names

This section describes how to define symbolic host names, 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 host name. To define a symbolic host name, 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 host name.

Protocol Translation and Processing PAD Calls

This section explains how Cisco routers initiate and accept PAD calls using protocol translation.

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 are 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 normally are used to configure PAD and protocol translation access. They are also used to override the configuration of the interface on a per-destination basis.

The following example configures an X.25 interface to restrict incoming PAD access to a single mapped host. This example requires that both incoming and outgoing PAD access 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.
- Address/subaddress matches the address for the router set by the **x25 host** command.

Accepting Incoming PAD Protocol Translation Calls

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 CUD field that pattern 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 and 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 You can specify the CUD field in the **translate** command in ASCII or octal. You cannot enter CUD values in hexadecimal in the **pad** or **translation** command. However, you can enter the octal equivalents of CUD hexadecimal values using the following command syntax:

```
pad x121-address /cud \307\021
```

or

```
translate x25 x121-address cud \307\021 tcp ip-address
```

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 the 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 user data 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 up to 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, beginning in global configuration mode:

Command	Purpose
`Router(config)# line vty line-number`	Increases the number of virtual terminal lines, and thus the maximum number of protocol translation sessions.
`Router(config-line)# no line vty line-number`	Decreases the number of virtual terminal lines, and thus the maximum number of protocol translation sessions.

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 8. One virtual terminal is required for each protocol translation session.

Table 8 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).

Enabling Asynchronous Functions on Virtual Terminal Lines

Using Cisco IOS software, you can configure asynchronous protocol features such as PPP and SLIP on virtual terminal lines. PPP and SLIP normally function only on asynchronous interfaces, not on virtual terminal lines. When you configure a vty to support asynchronous protocol features, you are creating *virtual asynchronous interfaces* on the virtual terminal lines. One practical benefit of virtual asynchronous interfaces is the ability to tunnel PPP and SLIP across X.25, TCP, or LAT networks on virtual terminal lines. You tunnel PPP and SLIP using the protocol translation facility.

To configure and use virtual asynchronous interfaces, perform the tasks described in the following sections:

- Creating Virtual Asynchronous Interfaces (Required)
- Enabling Protocol Translation of PPP and SLIP on Virtual Asynchronous Interfaces (Optional)
- Enabling IPX-PPP over X.25 to an IPX Network on Virtual Terminal Lines (Optional)
- Enabling Dynamic Routing on Virtual Asynchronous Interfaces (Optional)
- Enabling TCP/IP Header Compression on Virtual Asynchronous Interfaces (Optional)
- Enabling Keepalive Updates on Virtual Asynchronous Interfaces (Optional)
- Setting an MTU on Virtual Asynchronous Interfaces (Optional)
- Enabling PPP Authentication on Virtual Asynchronous Interfaces (Optional)

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

Creating Virtual Asynchronous Interfaces

To create a virtual asynchronous interface, use the following command in global configuration mode:

Command	Purpose
`Router(config)# vty-async`	Configures all virtual terminal lines to support asynchronous protocol features.

Enabling Protocol Translation of PPP and SLIP on Virtual Asynchronous Interfaces

One practical benefit of enabling virtual asynchronous interfaces is the ability to tunnel PPP and SLIP over X.25, thus extending remote node capability into the X.25 area. You can also tunnel PPP and SLIP over Telnet or LAT on virtual terminal lines. You can tunnel PPP and SLIP over X.25, LAT, or Telnet, but you do so by using the protocol translation feature in the Cisco IOS software.

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

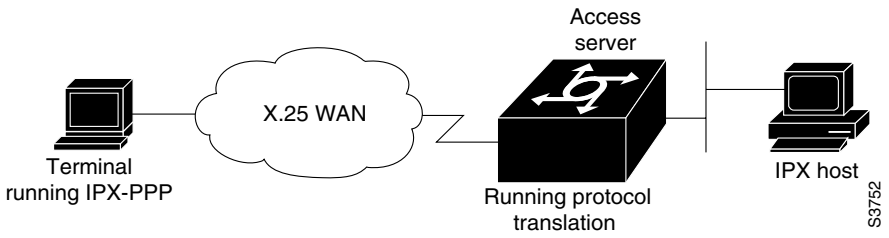
- If you are tunneling SLIP or PPP using the one-step method, you need not enter the **vty-async** command. Using the **translate** command with the **slip** or **ppp** keyword for one-step connections automatically enables asynchronous protocol functions on a per-vty basis.
- If you are tunneling SLIP or PPP using the two-step method, you must first enter the **vty-async** command on a global basis. Next, you perform a two-step connection process.

Enabling IPX-PPP over X.25 to an IPX Network on Virtual Terminal Lines

You can enable IPX-PPP on virtual terminals, which permits clients to log in to a virtual terminal on a router, invoke a PPP session at the EXEC prompt to a host, and run IPX to the host.

For example, in Figure 5 the client terminal on the X.25 network logs in to the vty on the access server, which is configured for IPX-PPP. When the user connects to the access server and the EXEC prompt appears, the user issues the PPP command to connect to the IPX host. The virtual terminal is configured to run IPX, so when the PPP session is established from the access server, the terminal can access the IPX host using an IPX application.

Figure 5 IPX-PPP on a Virtual Asynchronous Interface



To enable IPX to run over your PPP sessions on virtual terminal lines, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# ipx routing [node]`	Enables IPX routing.
Step 2	`Router(config)# interface loopback number`	Creates a loopback interface.
Step 3	`Router(config-if)# ipx network network`[1]	Enables a virtual IPX network on the loopback interface.
Step 4	`Router(config-if)# vty-async ipx ppp-client loopback number`	Enables IPX-PPP on virtual terminal lines by assigning the virtual terminal to the loopback interface configured for IPX.

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

Enabling Dynamic Routing on Virtual Asynchronous Interfaces

To route IP packets using the Interior Gateway Routing Protocol (IGRP), RIP, and OSPF routing protocols on virtual asynchronous interfaces, use the following command in global configuration mode:

Command	Purpose
`Router(config)# vty-async dynamic-routing`	Enables dynamic routing of IP packets on all virtual terminal lines.

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

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

Enabling TCP/IP Header Compression on Virtual Asynchronous Interfaces

You can compress the headers on TCP/IP packets on virtual asynchronous interfaces to reduce their size and increase performance. This feature only compresses the TCP header, so it has no effect on UDP packets or other protocol headers. The TCP header compression technique, described fully in RFC 1144, is supported on virtual asynchronous interfaces using PPP and SLIP encapsulation. You must enable compression on both ends of the connection.

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

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

Command	Purpose
`Router(config)# vty-async header-compression [passive]`	Enables header compression on IP packets on all virtual terminal lines.

Enabling Keepalive Updates on Virtual Asynchronous Interfaces

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

Command	Purpose
`Router(config)# vty-async keepalive seconds`	Specifies the frequency with which the Cisco IOS software sends keepalive messages to the other end of an asynchronous serial link.

The default interval is 10 seconds. It is adjustable in 1-second increments from 0 to 32,767 seconds. To turn off keepalive updates, set the value to 0. A connection is declared down after three update intervals have passed without a keepalive packet being received.

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

Setting an MTU on Virtual Asynchronous Interfaces

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

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

For example, at 9600 baud a 1500-byte packet takes about 1.5 seconds to transmit. This delay would indicate an MTU size of about 200, as derived from the following equations:

1.5 seconds / 0.2 seconds = 7.5

1500-byte packet / 7.5 = 200-byte packet

To specify the maximum IP packet size, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# vty-async mtu` *bytes*	Specifies the size of the largest IP packet that the virtual asynchronous interface can support.

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

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

Enabling PPP Authentication on Virtual Asynchronous Interfaces

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

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

Enabling CHAP

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

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

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

The required response has two parts:

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

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

Because this response is sent, the secreted information is never sent, thus preventing other devices from stealing it and gaining illegal access to the system. Without the proper response, the remote device cannot connect to the local router.

CHAP transactions occur only when a link is established. The local router does not request a password during the rest of the session. (The local router can, however, respond to such requests from other devices during a session.)

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

Command	Purpose
`Router(config)# vty-async ppp authentication chap`	Enables CHAP on all virtual asynchronous interfaces.

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

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

Enabling PAP

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

To enable PAP, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)# vty-async ppp authentication pap`	Enables PAP on all virtual asynchronous interfaces.

Enabling PPP Authentication via TACACS on Virtual Asynchronous Interfaces

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

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

Command	Purpose
`Router(config)# vty-async ppp use-tacacs`	Enables TACACS on all virtual asynchronous interfaces.

Maintaining Virtual Interfaces

To maintain virtual interfaces, perform the tasks described in the following sections:

- Monitoring and Maintaining a Virtual Access Interface
- Displaying a Virtual Asynchronous Interface
- Troubleshooting Virtual Asynchronous 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, and is the only way a virtual access interface can be created. However, a virtual access interface can be cleared and displayed.

To display or clear a specific virtual access interface, use any the following commands in 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 **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 = 272.22.213.7
  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 the 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, the Cisco IOS software displays all messages to the console terminal.

To monitor protocol translation connections, perform the tasks described in the following sections:

- Logging vty-Asynchronous Authentication Information to the Console Terminal
- Logging vty-Asynchronous Authentication Information to a Buffer
- Logging vty-Asynchronous Authentication Information to a UNIX Syslog Server

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 the Cisco IOS command references for explanations of command output.

Virtual Template for Protocol Translation Examples

The following sections show examples of configuring tunneling of PPP and SLIP using one-step and two-step protocol translation:

One-Step Examples

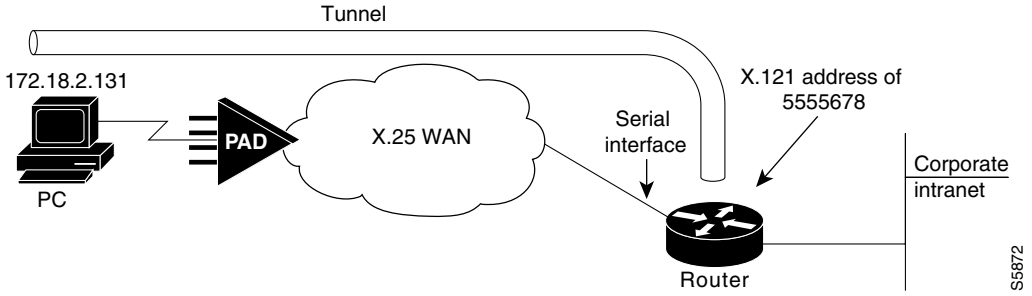
The examples in the following sections show how to configure virtual templates and apply them in one-step protocol translation sessions:

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 6. 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 6.

```
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 6 Tunneling PPP Across an X.25 Network



Tunnel SLIP Across X.25 Example

The following example uses SLIP encapsulation instead of the 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 Specifying 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, 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

The examples in the following sections show how to create and configure virtual interface templates and apply them in two-step protocol translation sessions:

- Two-Step Tunneling of PPP with Dynamic Routing and Header Compression Example
- Two-Step Tunneling of PPP with Dynamic Routing, TACACS, and CHAP Example

Two-Step Tunneling of PPP with Dynamic Routing and Header Compression Example

The following example uses the default PPP encapsulation on the virtual template. The example does not specify a peer default IP address because it is using two-step translation.

```
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

This section provides protocol translation examples for the following scenarios:

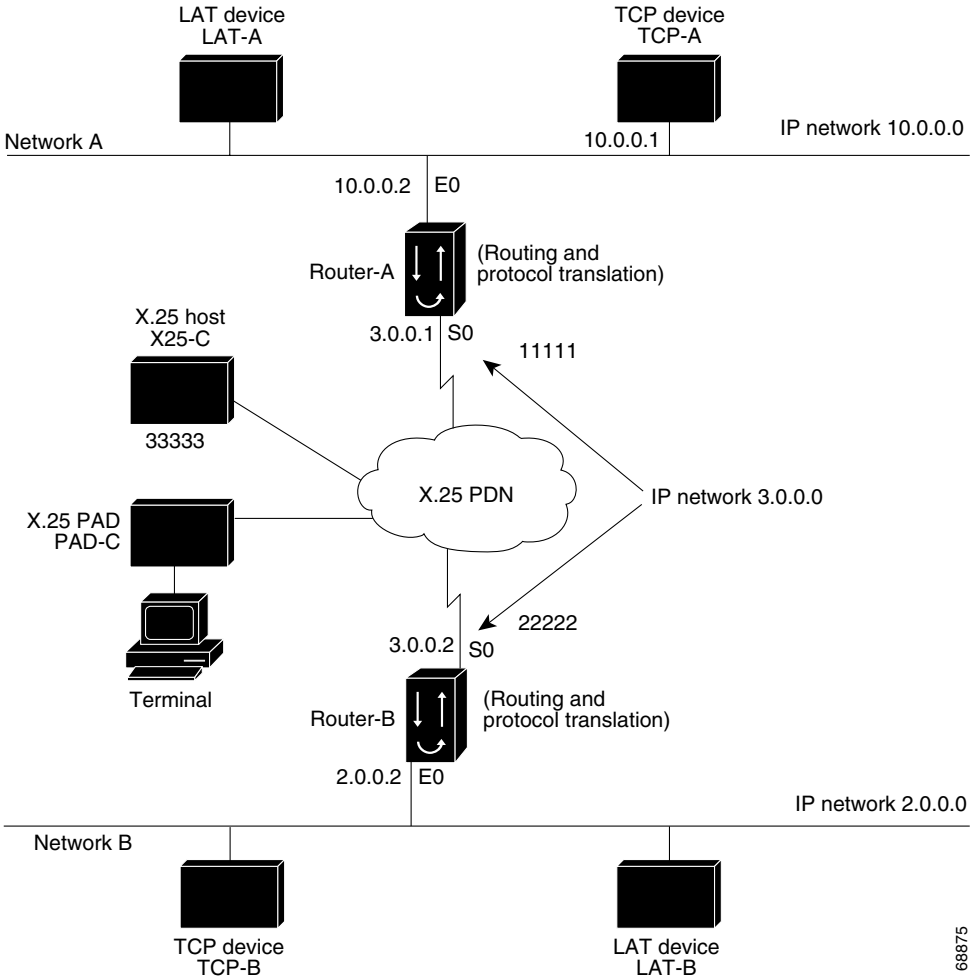
- Basic Configuration Example
- Central Site Protocol Translation Example
- Decreasing the Number of Translation Sessions Example
- Increasing the Number of Translation Sessions Example
- LAT-to-LAT over an IP WAN Example
- LAT-to-LAT over Frame Relay or SMDS Example
- LAT-to-LAT Translation over a WAN Example
- LAT-to-LAT over an X.25 Translation Example
- LAT-to-TCP Translation over a WAN Example
- LAT-to-TCP over X.25 Example
- LAT-to-X.25 Host Configuration Example
- Local LAT-to-TCP Translation Example
- Local LAT-to-TCP Configuration Example
- Standalone LAT-to-TCP Translation Example
- Tunneling SLIP Inside TCP Example
- Tunneling PPP over X.25 Example
- X.25 to L2F PPP Tunneling Example
- Assigning Addresses Dynamically for PPP Example
- Local IP Address Pool Example
- X.29 Access List Example
- X.3 Profile Example
- X.25 PAD-to-LAT Configuration Example
- X.25 PAD-to-TCP Configuration Example

Note In the application illustrations throughout the remainder of this chapter, source and destination device icons used to illustrate the flow of translated information are shown with black type in outlined shapes. Other elements in the environment are shown with reverse type on solid black shapes.

Basic Configuration Example

The following examples illustrate the basic global configuration commands and interface configuration commands for setting up Router-A (connected to Network A) and Router-B (connected to Network B), as illustrated in Figure 7. Refer to the chapter "Configuring Dial-In Terminal Services," for more information about LAT. For information on configuring X.25, refer to the *Cisco IOS Wide-Area Networking Configuration Guide*, Release 12.2.

Figure 7 Routers with Protocol Translation



Note The examples that follow focus on creating configurations that support one-step protocol translation. These connections can also be made using the two-step protocol translation method.

Configuration for Router-A

The following partial configuration for Router-A outlines a baseline configuration for Ethernet and serial interfaces on a router and configures support for IP, LAT, and X.25:

```
interface ethernet 0
 ip address 10.0.0.2 255.255.0.0
 !
 ! Enable LAT on interface.
 lat enabled
 !
interface serial 0
 encapsulation X.25
 x25 address 11111
 !
 ! The following parameters may depend on your network.
 x25 facility packetsize 512 512
 x25 facility windowsize 7 7
 !
  ! IP address and MAP command needed only if routing IP.
 ip address 10.3.0.1 255.255.0.0
 x25 map ip 10.3.0.2 22222 broadcast
 !
! Set up IP routing.
router igrp 100
 network 10.0.0.0
 network 10.3.0.0
 !
 ! Advertise as available for connections via LAT.
 ! Use this name (router-A) if connecting via 2-step method
 ! (for connecting directly to a specific router).
 lat service router-A enable
 !
 ! Set up some IP host names/addresses.
 ip host router-A 10.0.0.2 10.3.0.1
 ip host TCP-A 10.0.0.1
 ip host TCP-B 10.2.0.1
 ip host router-B 10.3.0.2 10.2.0.2
```

Configuration for Router-B

The following partial configuration for Router-B outlines a baseline configuration for Ethernet and serial interfaces on a router and configures support for IP, LAT, and X.25:

```
interface ethernet 0
 ip address 10.2.0.2 255.255.0.0
 !
 ! Enable LAT on interface.
 lat enabled
 !
interface serial 0
 encapsulation X.25
 x25 address 22222
 ! The following parameters may depend on your network.
 x25 facility packetsize 512 512
 x25 facility windowsize 7 7
 !
 ! IP address and MAP command needed only if routing IP.
 ip address 10.3.0.2 255.255.0.0
 x25 map ip 10.3.0.1 11111 broadcast
 !
 ! Set up IP routing.
router igrp 100
```

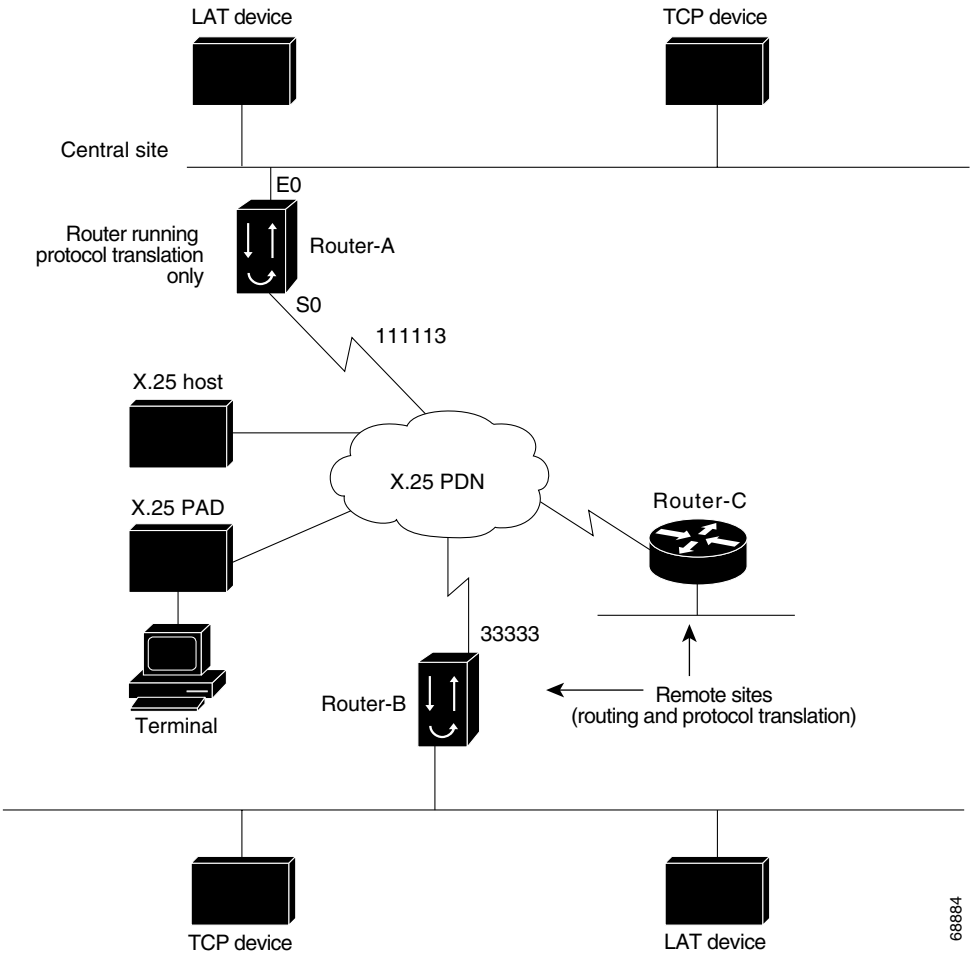
```
network 10.2.0.0
network 10.3.0.0
!
! Advertise as available for connections via LAT.
! Use this name (router-B) if connecting via 2-step method
! (for connecting directly to a specific router).
lat service router-B enable
!
! Set up some IP host names/addresses.
ip host router-A 10.3.0.1 10.0.0.2
ip host TCP-A 10.0.0.1
ip host TCP-B 10.2.0.1
ip host router-B 10.2.0.2 10.3.0.2
```

Note You can specify IP host names used to identify specific hosts by explicitly using the **ip host** global configuration command or by using Domain Name System (DNS) facilities.

Central Site Protocol Translation Example

To support central site protocol translation, a router with an image that supports protocol translation is directly connected back-to-back to another router (see Figure 8). This second device acts as an X.25 switch by sending X.25 packets to Router-B while concurrently routing and bridging other protocols.

Figure 8 Central Site Protocol Translation Example



The following example shows how to configure a router to support translating protocols over an X.25 network among multiple sites. Router-C is configured to act as an X.25 switch to send X.25 packets to Router-A while concurrently routing and bridging other protocols.

The following example also shows how to use the **translate** global configuration command to translate LAT and TCP over X.25 WAN media. In this configuration, Router-A can translate LAT or TCP traffic into X.25 packets for transmission over an X.25 PDN network. Packets are then translated back to LAT or TCP on the other side of the WAN.

```
interface ethernet 0
 ip address 10.0.0.2 255.255.0.0
 !
 ! Enable LAT on interface if concurrently routing (8.3 feature).
 lat enable
 !
interface serial 0
 encapsulation X.25
 ! Note that this is subaddress 3 of 11111.
 x25 address 111113
 ! The following parameters may depend on your network.
 x25 facility packetsize 512 512
 x25 facility windowsize 7 7
 no ip address
```

```
 ! Translate Configuration for router-A.
 !
 no ip routing
 ! Note subaddress 03 of address 111113.
translate x25 11111303 tcp tcpdevice
translate lat TCP-B x25 3333301
translate lat lat-device tcp tcp-device
! etc...any translate commands needed by application.
```

Decreasing the Number of Translation Sessions Example

The following example sets the number of protocol translation sessions to 10, whether routing is turned on or off:

```
no line vty 10
```

Increasing the Number of Translation Sessions Example

The following example sets the number of protocol translation sessions to 120, whether routing is turned on or off:

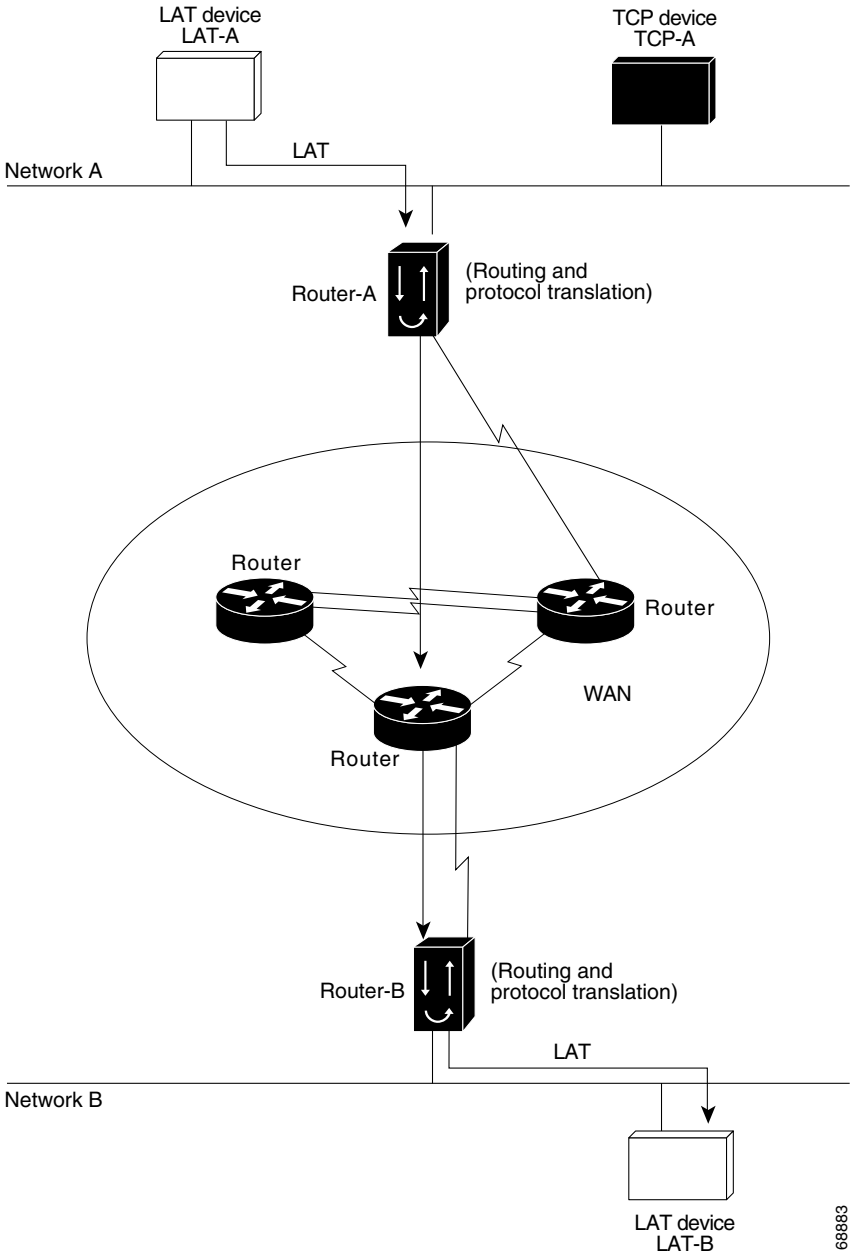
```
line vty 119
```

LAT-to-LAT over an IP WAN Example

The Cisco IOS software can be used to connect LAT devices over a WAN backbone that only allows routable protocols (see Figure 9). This configuration exists when LAT networks are either isolated or on their own internetwork.

With the protocol translation, LAT traffic can be translated to TCP and then routed on the WAN as TCP traffic. The LAT connections stay local between the LAT device and the router running the protocol translation option. Thus, connections are not susceptible to delays on the WAN. This capability reduces the amount of traffic on the WAN because only the data from specific LAT sessions is forwarded on the WAN rather than all the LAT protocol status information packets.

Figure 9 LAT-to-LAT over an IP WAN



The following example illustrates how to use the **translate** global configuration command to translate from LAT to LAT when an IP WAN is used. In this configuration, Router-B with the protocol translation option routes encapsulated packets translated from LAT to TCP over the WAN. Router-A translates packets back to LAT on the other side of the WAN. Example translation configurations for both Router-A and Router-B are shown, but these examples do not include configuration information for devices in the WAN.

The following examples are essentially the same configurations for protocol translation as those in the following Frame Relay example:

```
! Translate LAT to TCP/Telnet for Router-A, which is on Network A.
translate lat DISTANT-LAT tcp Router-A
```

```
! Translate TCP to LAT for Router-B, which is on Network B.
translate tcp Router-B lat LAT-B
```

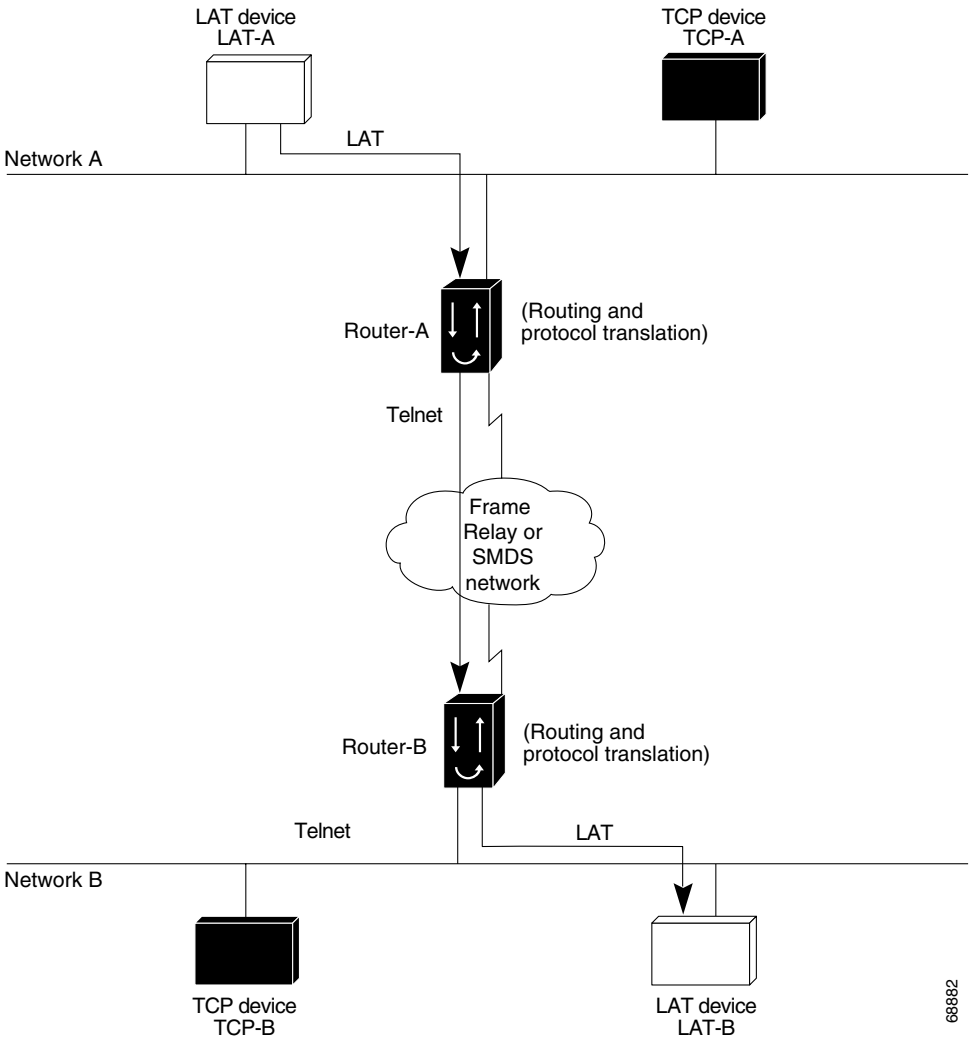
Note You can use the same name (for example, LAT-B) in the **translate** command for both Router-A and Router-B because each router operates independently. However, this symmetry is not required. The key is the common IP name in both **translate** commands.

LAT-to-LAT over Frame Relay or SMDS Example

To transport LAT traffic over a Frame Relay or an Switched Multimegabit Data Service (SMDS) network, LAT must first be translated to TCP. The TCP traffic is routed over the Frame Relay network and then translated back to LAT on Router-B on Network B (see Figure 10).

Note The interface configurations for a Frame Relay or an SMDS implementation differ from the specifications shown earlier in this chapter. For more information about configuring Frame Relay and SMDS, refer to the *Cisco IOS Wide-Area Networking Configuration Guide*, Release 12.2.

Figure 10 LAT-to-LAT over Frame Relay or SMDS



The following example illustrates how to use the **translate** global configuration command to translate from LAT to LAT when the WAN uses Frame Relay or SMDS. In this configuration, the Cisco IOS software routes encapsulated packets translated from LAT to TCP over the Frame Relay or SMDS network. Packets are then translated back to LAT on the other side of the Frame Relay or SMDS network.

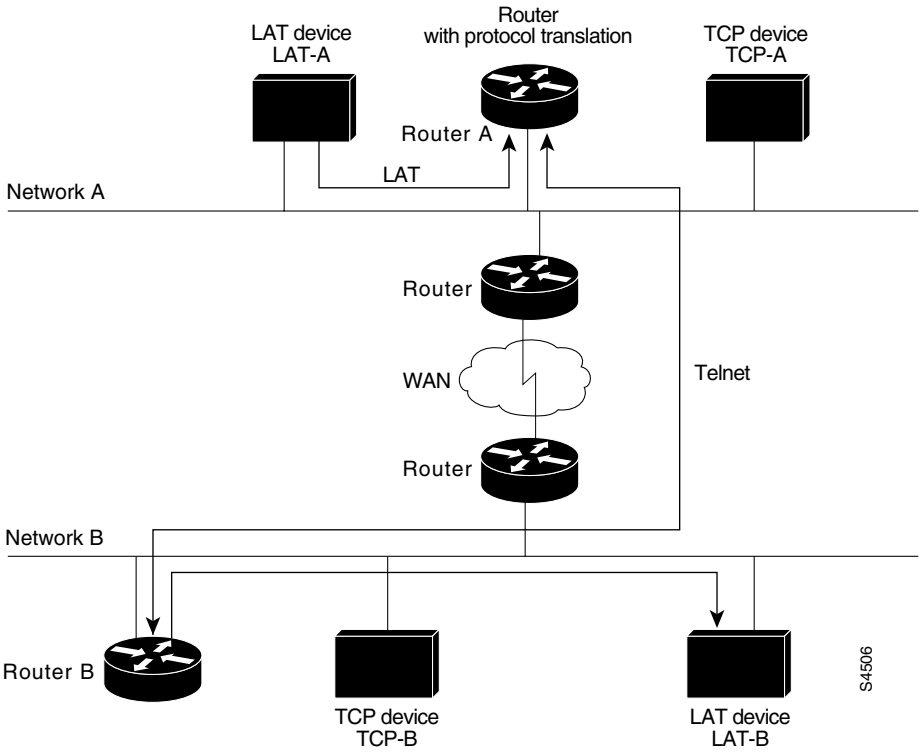
```
! Translate LAT to TCP/Telnet on router-A, which is on Network A.
translate lat DISTANT-LAT tcp router-A

! Translate TCP to LAT on router-B, which is on Network B.
translate tcp router-B lat LAT-B
```

Note You can use the same name (for example, LAT-B) in the **translate** command for both Router-A and Router-B because each router operates independently. However, this symmetry is not required. The key is the common IP name used in both **translate** commands.

LAT-to-LAT Translation over a WAN Example

In Figure 11, LAT can be transported to a remote LAT device by translating the packets to TCP format and using Telnet to send them across the WAN. The configuration files for the routers named Router-A and Router-B follow the figure. The logical name CS-B1 is the name given to device CS-B.

Figure 11 LAT-to-LAT Translation over a WAN



Configuration for Router-A

```
interface ethernet 0
 ip address 172.18.32.16 255.255.0.0
!
! Enable LAT on this interface.
 lat enabled
!
translate lat distant-LAT tcp TS-B1
```

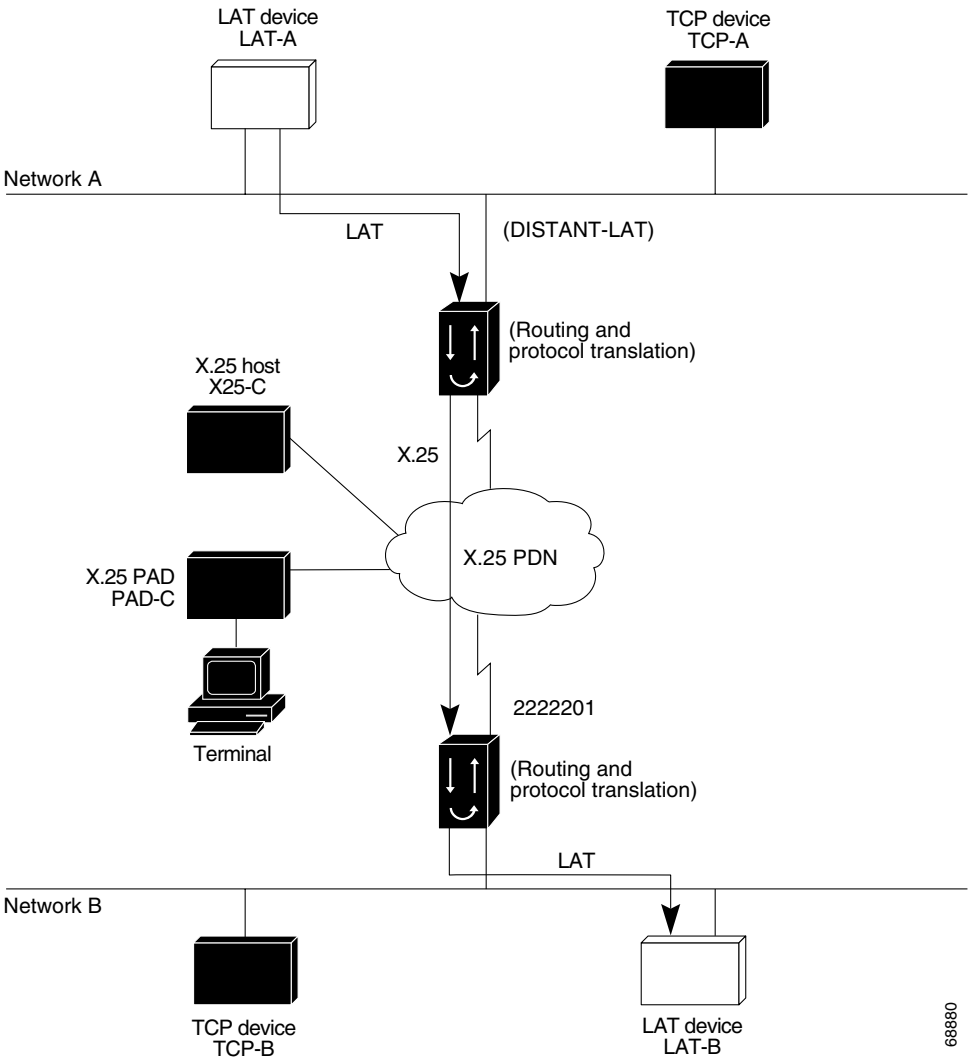
Configuration for Router-B

```
interface ethernet 0
 ip address 172.18.38.42 255.255.0.0
!
! Enable LAT on this interface.
 lat enabled
!
translate lat TS-B1 lat LAT-B
```

LAT-to-LAT over an X.25 Translation Example

Protocol translation provides transparent connectivity between LAT devices on different networks via an X.25 PDN. In Figure 12, which illustrates this application, the LAT device on Network A (LAT-A) first makes a virtual connection to the router named Router-A on Network A using the LAT protocol. Router-A then translates the LAT packets into X.25 packets and sends them through the X.25 network to Router-B on Network B. Router-B translates the X.25 packets back to LAT packets and establishes a virtual connection to the LAT device on Network B (LAT-B). These handoffs are handled transparently when the Cisco IOS software is configured for one-step protocol translation.

Figure 12 LAT-to-LAT via an X.25 PDN



The following example shows how to use the **translate** global configuration command to translate from LAT to X.25 and from X.25 back to LAT to allow connection service to a LAT device on Network B from a LAT device on Network A. This example requires two separate configurations, one for each LAT device.

```
! Translate LAT to X.25 on router-A, which is on Network A.
translate lat DISTANT-LAT x25 2222201
```

```
! Translate X.25 to LAT on router-B, which is on Network B.
translate x25 2222201 lat LAT-B
```

In the first **translate** command, DISTANT-LAT defines a LAT service name for Router-A. When a user on device LAT-A attempts to connect to LAT-B, the target specified in the **connect** command is DISTANT-LAT.

In the **translate** command for Router-B, the name of the LAT service on the target host (LAT-B) is LAT-B. Router-B translates the incoming X.25 packets from 2222201 to LAT and then transparently relays these packets to LAT-B.

The following example shows a connection request. When the user enters this command, a connection attempt from LAT-A on Network A to TCP-B on Network B is attempted.

```
Router> connect DISTANT-LAT
```

To configure Router-B to send information back from LAT-B to LAT-A, use commands symmetrical to the prior configuration (this path is not shown in Figure 12):

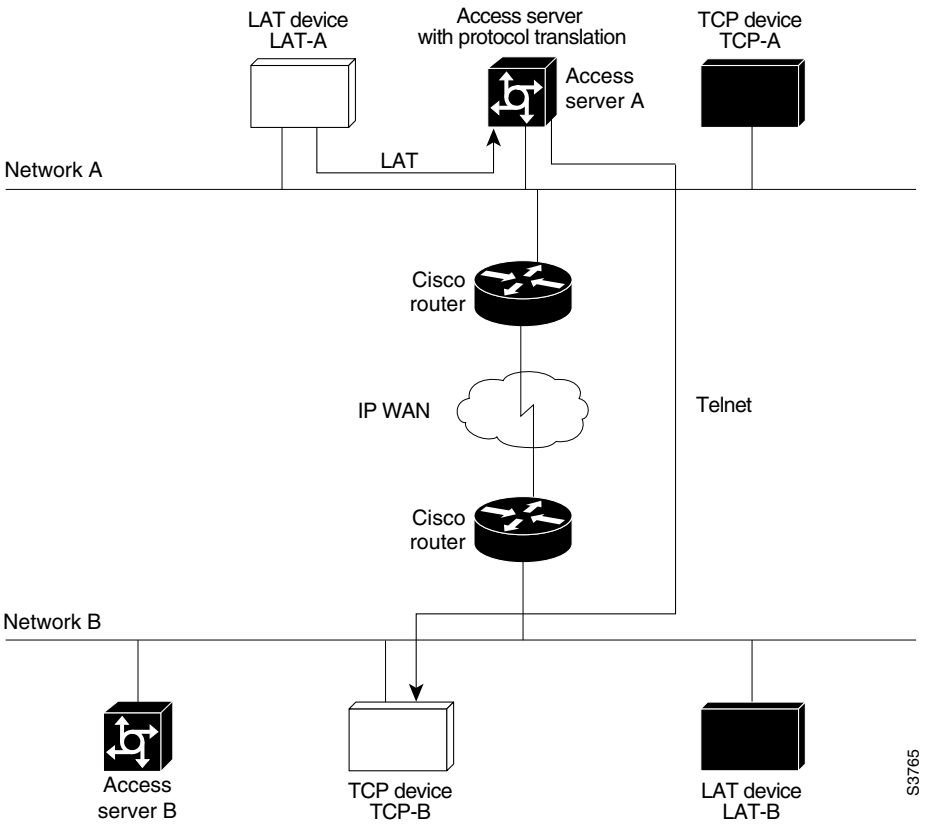
```
! Translate LAT to X.25 on router-B, which is on Network B.
translate lat FAR-LAT x25 1111103
! Translate X.25 to LAT on router-A, which is on Network A.
translate x25 1111103 lat LAT-A
```

Note You can use the same name (for example, LAT-B) in the **translate** command for both Router-A and Router-B because each router with the protocol translation option operates independently. However, this symmetry is not required. The key is the common X.121 address used in both **translate** commands. If you prefer to have unique service names, set the names in each router to be the same.

LAT-to-TCP Translation over a WAN Example

Figure 13 shows a configuration that allows translation of LAT to TCP and transmission across an IP-based WAN. The configuration file for the access server identified as A follows the figure. The logical LAT service name distant-TCP is the name given to device TCP-B.

Figure 13 LAT-to-TCP Translation over a WAN



Configuration for Access Server A

```
interface ethernet 0
 ip address 172.18.38.42 255.255.0.0
!
! Enable LAT on this interface.
 lat enabled
!
translate lat distant-TCP tcp TCP-B
```

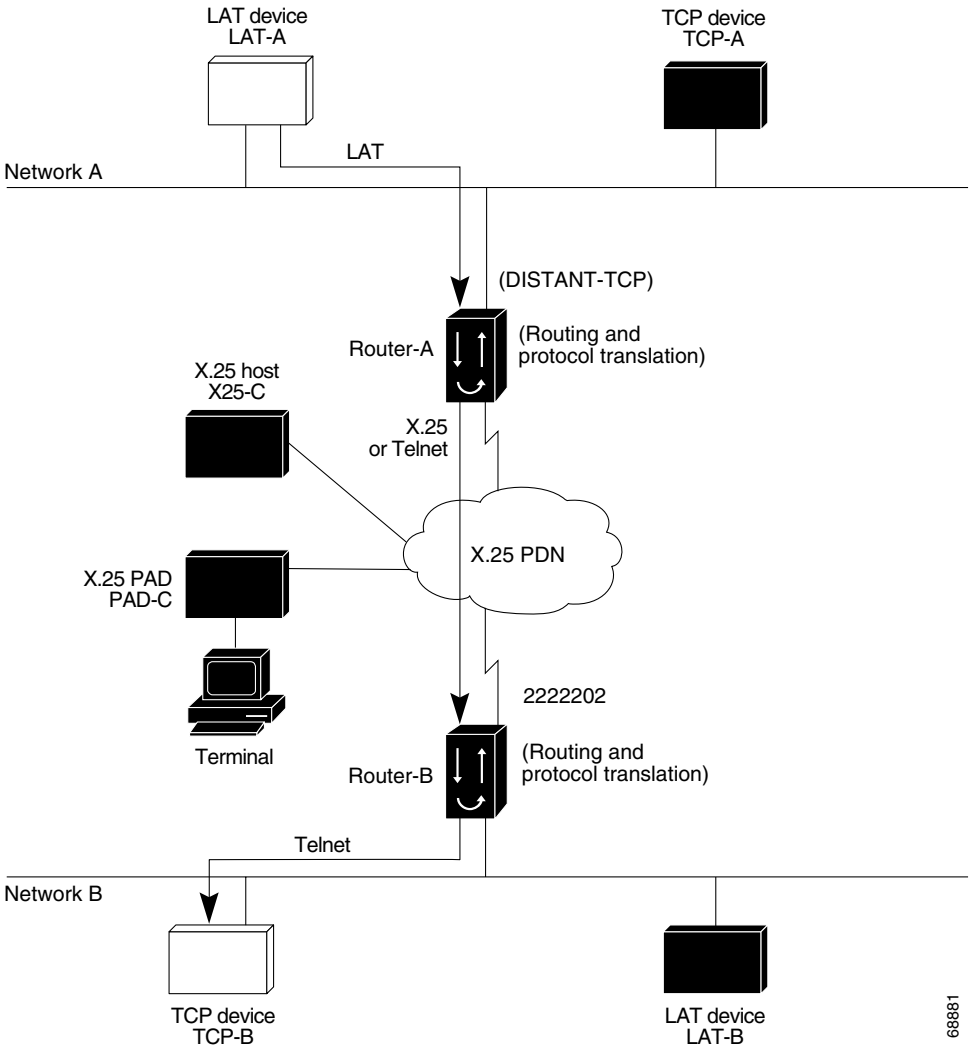
LAT-to-TCP over X.25 Example

You can use protocol translation to provide transparent connectivity between LAT and TCP devices on different networks via an X.25 PDN. In Figure 14, which illustrates this application, the LAT device on Network A is communicating with the TCP device on Network B. There are two ways to provide this connectivity: The LAT traffic from Network A can be translated into either X.25 packets, or TCP/IP packets can be sent out on the X.25 PDN.

If the traffic is translated from LAT directly into X.25 frames by Router-A, Router-B on Network B translates incoming packets intended for device TCP-B into TCP. If Router-A converts LAT to TCP, the TCP traffic is being encapsulated in X.25 and sent on the X.25 network. Router-B on Network B strips off the encapsulation and routes the TCP packet. In this case, protocol translation is not needed on Router-B.

If the traffic is translated to TCP by Router-A, the packets are encapsulated within X.25 frames. In general, translating the traffic directly to X.25 is more efficient in this application because no encapsulation is necessary. X.25 packets have only 5 bytes of header information, and TCP over X.25 has 45 bytes of header information.

Figure 14 LAT-to-TCP via X.25



The following example shows how to use the **translate** global configuration command to translate from LAT to X.25 (on Router-A) and from X.25 to TCP (on Router-B), thus allowing connection service to a TCP device on Network B (TCP-B) from a LAT device on Network A (LAT-A). You must configure Router-A and Router-B separately.

```
! Translate LAT to X.25 on router-A, which is on Network A.
translate lat DISTANT-TCP x25 2222202

! Translate X.25 to TCP on router-B, which is on Network B.
translate x25 2222202 tcp TCP-B
```

In the **translate** command for Router-A, DISTANT-TCP defines a LAT service name for Router-A. When a user on device LAT-A attempts to connect to LAT-B, the target specified in the **connect** command is DISTANT-TCP.

In the **translate** command for Router-B, the TCP service on the target host is TCP-B. Router-B translates the incoming X.25 packets from 2222202 to TCP packets and transparently relays these packets to TCP-B.

The following example shows a connection request. When the user enters this command, a connection attempt from LAT-A on Network A to LAT-B on Network B is attempted.

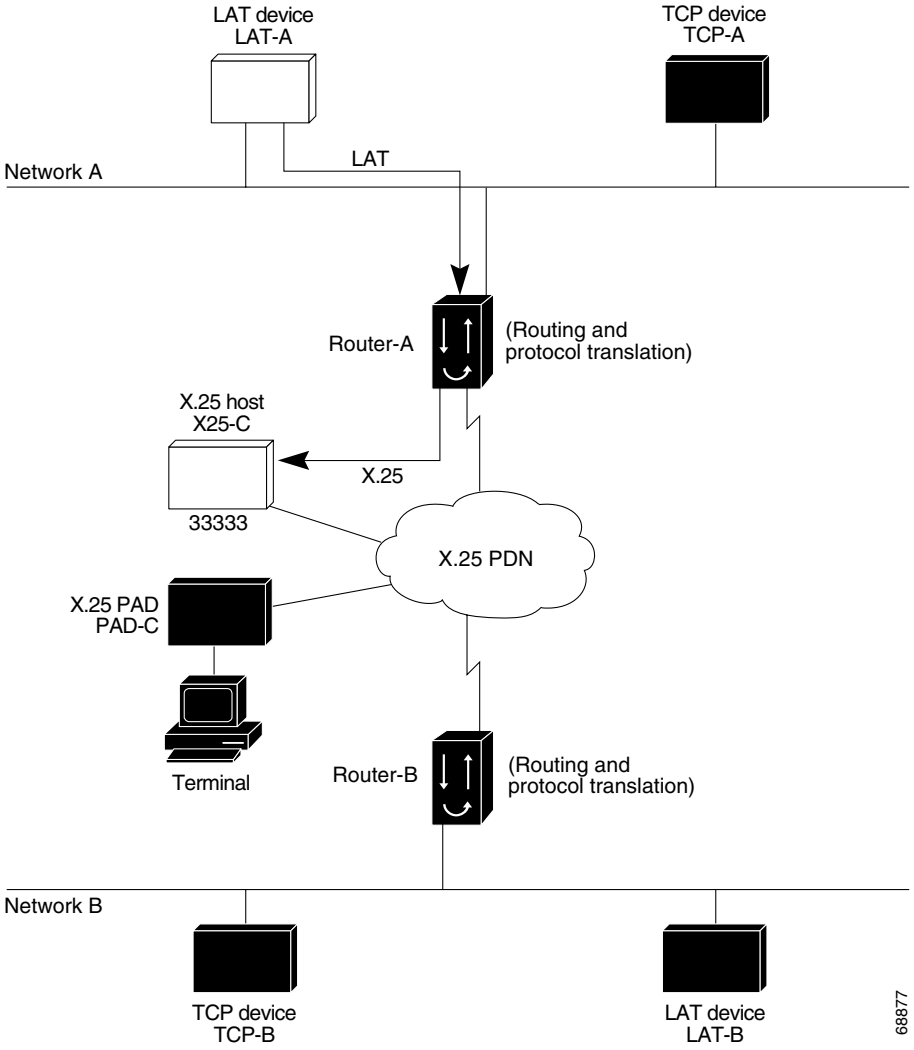
```
local> connect DISTANT-TCP
```

Note You can use the same name (for example, TCP-B) in the **translate** command for both Router-A and Router-B because each router operates independently. However, this symmetry is not required. The key is the common X.121 address used in both **translate** commands. If you prefer to have unique service names, set the names in each router to be the same.

LAT-to-X.25 Host Configuration Example

Figure 15 shows a protocol translation configuration that permits LAT devices to communicate with X.25 hosts through an X.25 PDN. In the application illustrated in Figure 15, LAT-A is a LAT device that is communicating with X25-C, an X.25 host. The LAT traffic from LAT-A is translated to X.25.

Figure 15 LAT-to-X.25 Host Translation



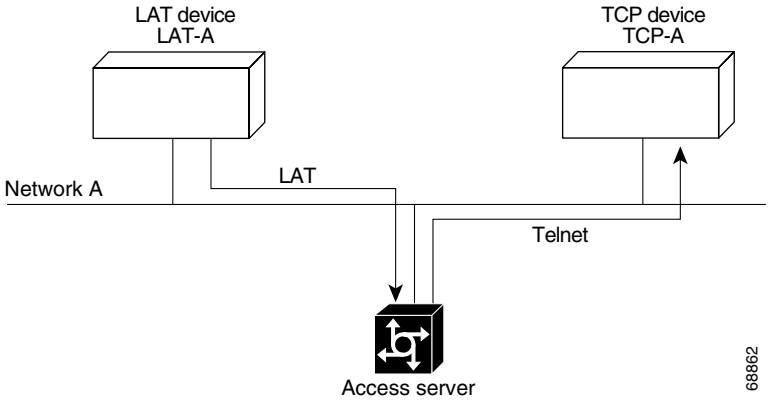
The following example shows how to use the **translate** global configuration command to translate from LAT to X.25. It is applied to Router-A. This example sets up reverse charging for connections, which causes the router with the protocol translation option to instruct the PDN to charge the destination for the connection. It is essentially a collect call. The reversal of charges must be prearranged with the PDN and destination location (on an administrative basis), or the call will not be accepted.

```
! Translate LAT to X.25 host, with reverse charging.
translate lat X25-C x25 33333 reverse
!
! Specify optional X.25 hostname.
x25 host X25-C 33333
```

Local LAT-to-TCP Translation Example

Figure 16 shows a simple LAT-to-TCP translation across an Ethernet network. Its Cisco IOS configuration file follows the figure. The name TCP-A is the logical name given to the device TCP-A.

Figure 16 Local LAT-to-TCP Translation



Configuration for the Access Server

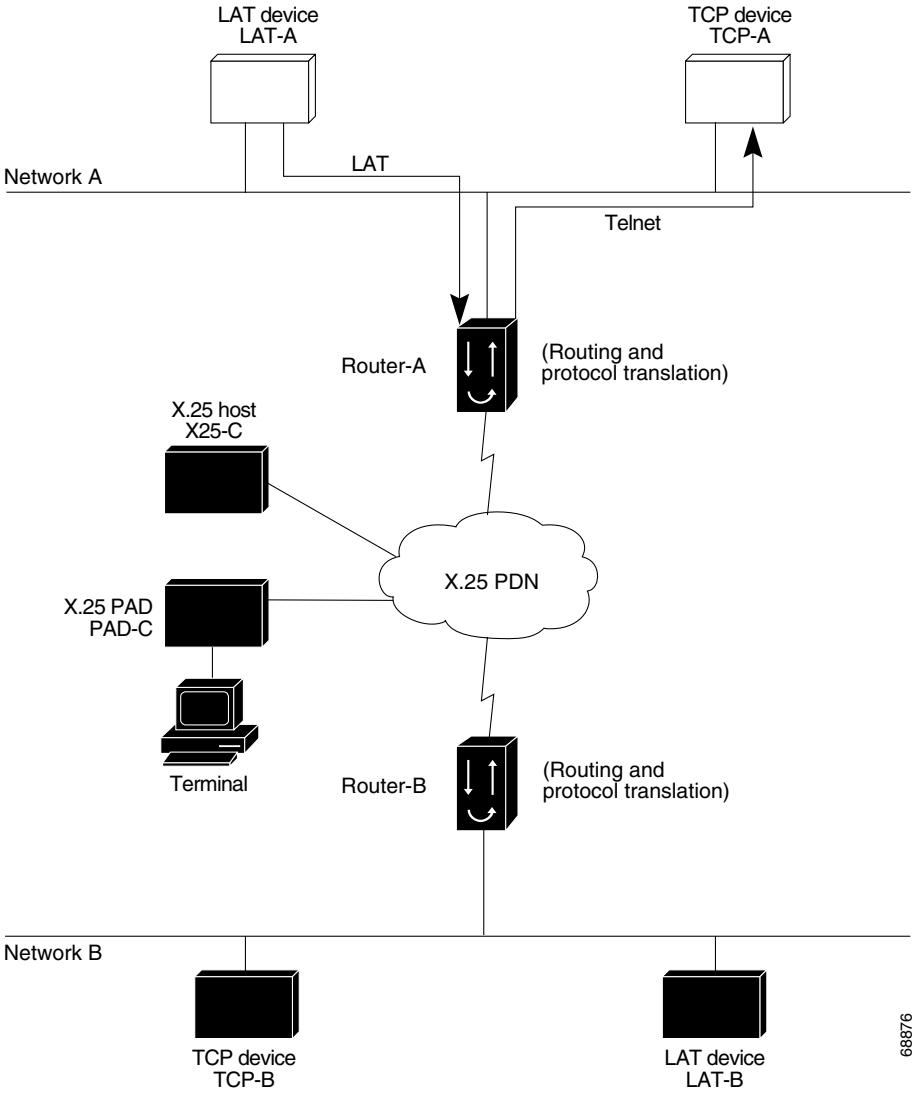
```
interface ethernet 0
 ip address 172.18.38.42 255.255.0.0
!
! Enable LAT on this interface.
 lat enabled
!
translate lat TCPA tcp TCP-A
```

Local LAT-to-TCP Configuration Example

The Cisco IOS software running protocol translation can translate between LAT and Telnet traffic to allow communication among resources in these protocol environments. In Figure 17, the LAT device on Network A (LAT-A) is shown connecting to a device running Telnet (TCP-A).

The commands in this example are only part of the complete configuration file for an individual device.

Figure 17 Local LAT-to-TCP Translation



The following example configures Router-A to translate from LAT to TCP:

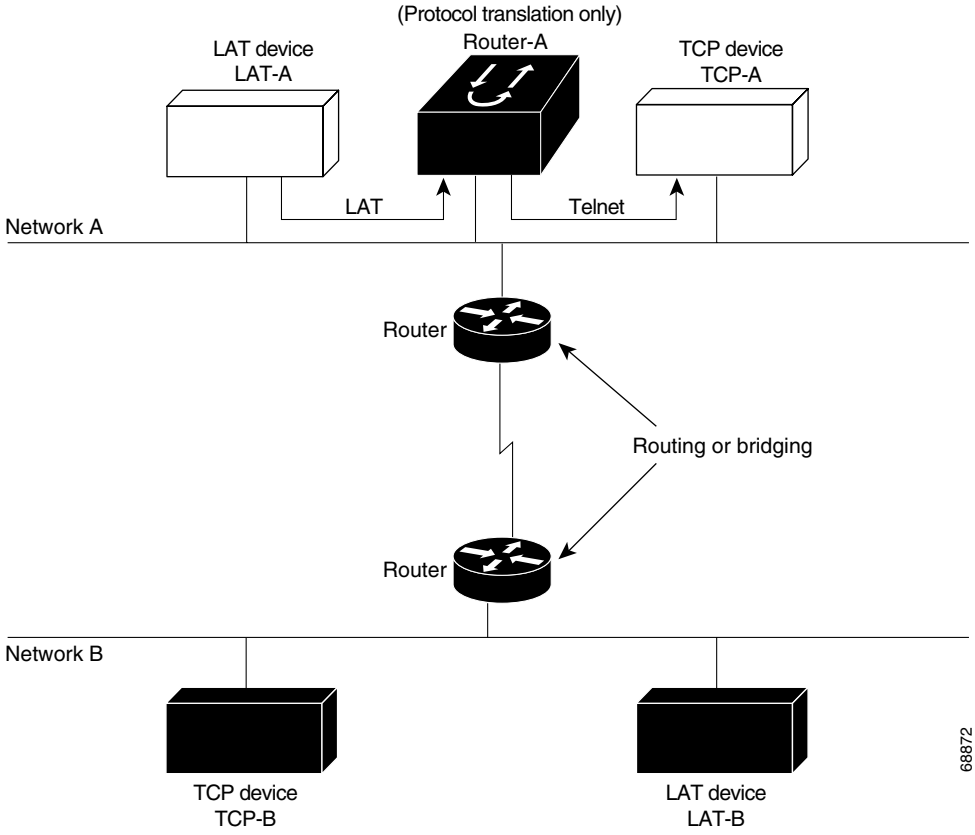
```
! Translate LAT connections to TCP for connectivity to TCP-A.
translate lat TCP-A tcp TCP-A
! Optional additional commands.
lat service TCP-A ident Protocol Translation to TCP-A
```

In the last command, the text string “Protocol Translation to TCP-A” is an identification string for the LAT service named TCP-A. This string is sent to other routers on the local network.

Standalone LAT-to-TCP Translation Example

If you need a large number of local LAT-to-TCP translation sessions, you can set up the router named Router-A to use only an Ethernet port, as the example following Figure 18 indicates. This application allows 100 concurrent translation sessions. In the applications shown in Figure 18, any other router that supports protocol translation can be used to interconnect network segments performing bridging or routing.

Figure 18 Router Functioning as a Standalone Protocol Translator



Configuration for Router-A

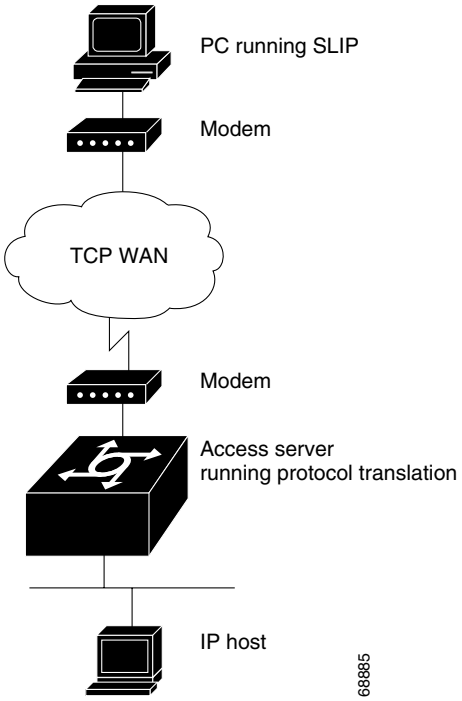
```
! Translation Configuration for Router-A only.
!
interface ethernet 0
 ip address 10.0.0.2 255.255.0.0
 !
 ! Enable LAT on this interface.
 lat enabled
 !
interface serial 0
 shutdown
 no ip routing
 default-gateway 10.0.0.100
!
translate lat TCP-A tcp TCP-A
translate lat TCP-B tcp TCP-B
translate tcp LAT-A lat lat-z
```

```
! etc...translate commands as required.
```

Tunneling SLIP Inside TCP Example

Protocol translation enables you to tunnel from TCP to SLIP to allow communication among resources in these protocol environments. In Figure 19, the PC running SLIP is connecting to a TCP/IP network and making a connection with the device IP host. The example following Figure 19 enables routing and turns on header compression.

Figure 19 Tunneling SLIP Inside TCP



The configuration tunnels SLIP inside of TCP packets from the SLIP client with IP address 10.2.0.5 to the router. It then establishes a protocol translation session to the IP host. Routing and header compression are enabled for the SLIP session.

```
translate tcp 10.0.0.1 slip 10.2.0.5 routing header-compression passive
```

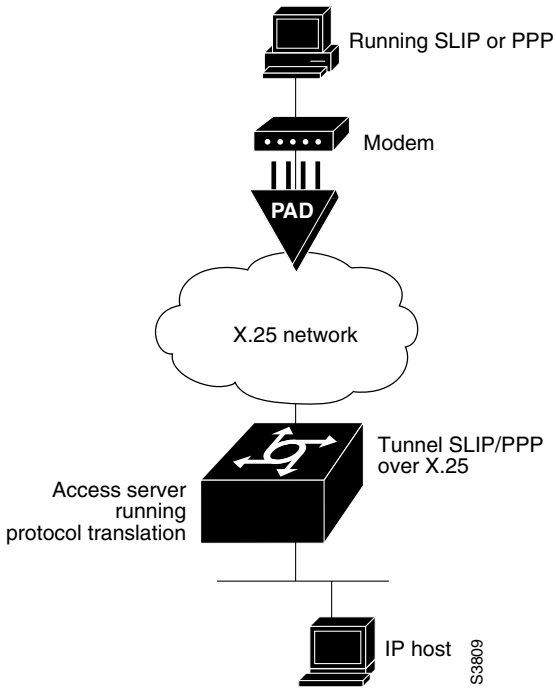
The device IP host on a different network attached to the router can be accessed by the SLIP client because routing has been enabled on the interface in the router where the SLIP session is established.

This example is incomplete. The commands in this example are only part of the complete configuration file for an individual router.

Tunneling PPP over X.25 Example

Cisco IOS software can tunnel PPP traffic across an X.25 WAN to allow communication among resources in these protocol environments. In Figure 20, the PC establishes a dialup PPP session through an X.25 network using CHAP authentication.

Figure 20 Tunneling PPP in X.25



The following configuration tunnels PPP over X.25 from the PPP client to the virtual asynchronous interface with IP address 10.0.0.4. Routing and CHAP authentication are enabled for the PPP session. The X.121 address of the X.25 host is 31370054065. An X.29 profile script named x25-ppp is created using the following X.3 PAD parameters:

1:0, 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, 17:24, 18:18, 19:0, 20:0, 21:0, 22:0

For more information about X.3 PAD parameters, refer to the appendix "X.3 PAD Parameters" at the end of this publication. If you were performing a two-step connection, you would specify these X.3 PAD parameters using the **pad** [**/profile** *name*] command.

With the router connected to the IP host, the PC running PPP can now communicate with the IP host.

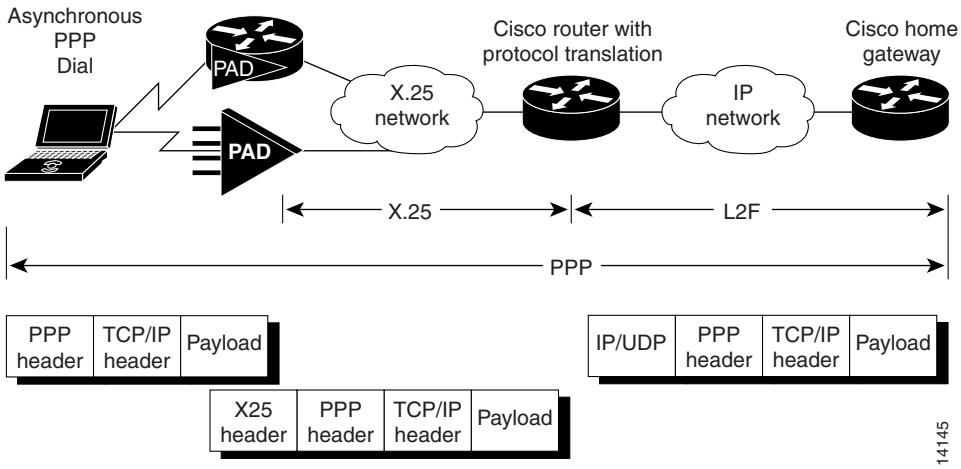
```
Router# configure terminal
Router(config)# X29 profile x25-ppp 1:0  2:0  3:2  4:1  5:0  6:0  7:21  8:0  9:0 10:0
11:14  12:0  13:0  14:0  15:0  16:127  17:24 18:18
Router(config)# translate x25 31370054065 profile x25-ppp ppp 10.0.0.4 routing
authentication chap
```

This example is incomplete. The commands in this example are only a part of the complete configuration file for an individual router.

X.25 to L2F PPP Tunneling Example

Protocol translation permits remote PPP users to connect to an X.25 PAD to communicate with IP network users via an L2F tunnel. (See Figure 21.)

Figure 21 L2F PPP Tunneling in X.25



The client application generates TCP/IP packets, which the PPP driver on the remote PC sends to the PAD. The PAD can either be an existing X.25/X.3/X.28/X.29-compliant PAD or a Cisco router with X.25 and PAD capability. The PAD receives the PPP/TCP/IP packets and sends them as X.25/PPP/TCP/IP packets to the X.25 network.

The Cisco router receives the packets and uses the protocol translation code to strip off the X.25 header. The router, using virtual templates, configures VPDN. VPDN invokes L2F tunneling and the virtual access interface via protocol translation, enables PPP to tunnel to the far home gateway and be terminated. At this point, the PC user can use Telnet, File Transfer Protocol (FTP), or similar file transfer utilities. The following is a partial example:

```
Router# virtual-temp 1
Router# encap ppp
Router# authentication chap
Router# trans x25 1234 virtual-temp 1
```

The following example shows a VPDN over a protocol translation virtual terminal-asynchronous connection over X.25 WAN. The client username is pc-user@cisco.com, the network access server is shadow (a Cisco router with the protocol translation option), and the home gateway is enkidu. The domain is cisco.com. The configuration for network access server shadow is as follows:

```
! VPDN NAS and Home Gateway passwords
username shadow password 7 013C142F520F
username enkidu-gw password 7 022916700202
vpdn enable
! VPDN outgoing to Home Gateway
vpdn outgoing cisco.com shadow ip 10.4.4.41
!
interface Virtual-Template1
 ip unnumbered Ethernet0
 no ip mroute-cache
 ppp authentication chap
!
interface Serial0
 description connects to enkidu s 0
 encapsulation x25 dce
 x25 address 2194440
 clockrate 2000000
!
translate x25 21944405 virtual-template 1
!
```

The configuration for home gateway enkidu-gw is as follows:

```
! VPDN NAS and Home Gateway passwords
 username shadow-nas password 7 143800200500
 username enkidu-gw password 7 132A05390208
 !
 ! The client user name and password
 username pc-user@cisco.com password 7 032B49200F0B
 !
 vpdn enable
 ! VPDN incoming from Shadow to this Home Gateway
 vpdn incoming shadow enkidu-gw virtual-template 1
 !
```

Assigning Addresses Dynamically for PPP Example

The following example shows how to configure the Cisco IOS software to assign an IP address dynamically to a PPP client using the one-step protocol translation facility:

```
! Enable DHCP proxy-client status on the router.
ip address-pool dhcp-proxy-client
! Specify rockjaw as the DHCP server on the network.
ip dhcp-server rockjaw
translate x25 5467835 ppp ip-pool keepalive 0
```

Local IP Address Pool Example

The following example shows how to select the IP pooling mechanism and how to create a pool of local IP addresses that are used when a client dials in on an asynchronous line. The address pool is named group1 and consists of interfaces 0 through 5.

```
! Tell the server to use a local pool.
ip address-pool local
! Define the range of ip addresses on the local pool.
ip local pool group1 172.18.35.1 192.168.35.5
translate x25 5467835 ppp ip-pool scope-name group1
```

X.29 Access List Example

The following example shows how to create an X.29 access list. Incoming permit conditions are set for all IP hosts and LAT nodes that have specific characters in their names. All X.25 connections to a printer are denied. Outgoing connections are restricted.

```
! Permit all IP hosts and LAT nodes beginning with "VMS".
! Deny 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 vty 5
 access-class 1 in
 !
 ! Permit outgoing connections for other lines.
 !
 ! Permit IP access with the network 172.16.
 access-list 2 permit 172.16.0.0 0.0.255.255
 !
```

```
 ! Permit LAT access to the prasad/gopala complexes.
 lat access-list 2 permit ^prasad$
 lat access-list 2 permit ^gopala$
 !
 ! Permit X.25 connections to Infonet hosts only.
 x29 access-list 2 permit ^31370
 !
line vty 0 16
 access-class 2 out
 !
translate tcp 172.16.1.26 x25 5551234 access-class 2
```

X.3 Profile Example

The following profile script turns local edit mode on when the connection is made and establishes local echo and line termination upon receipt of a Return character. The name linemode is used with the **translate** command to effect use of this script.

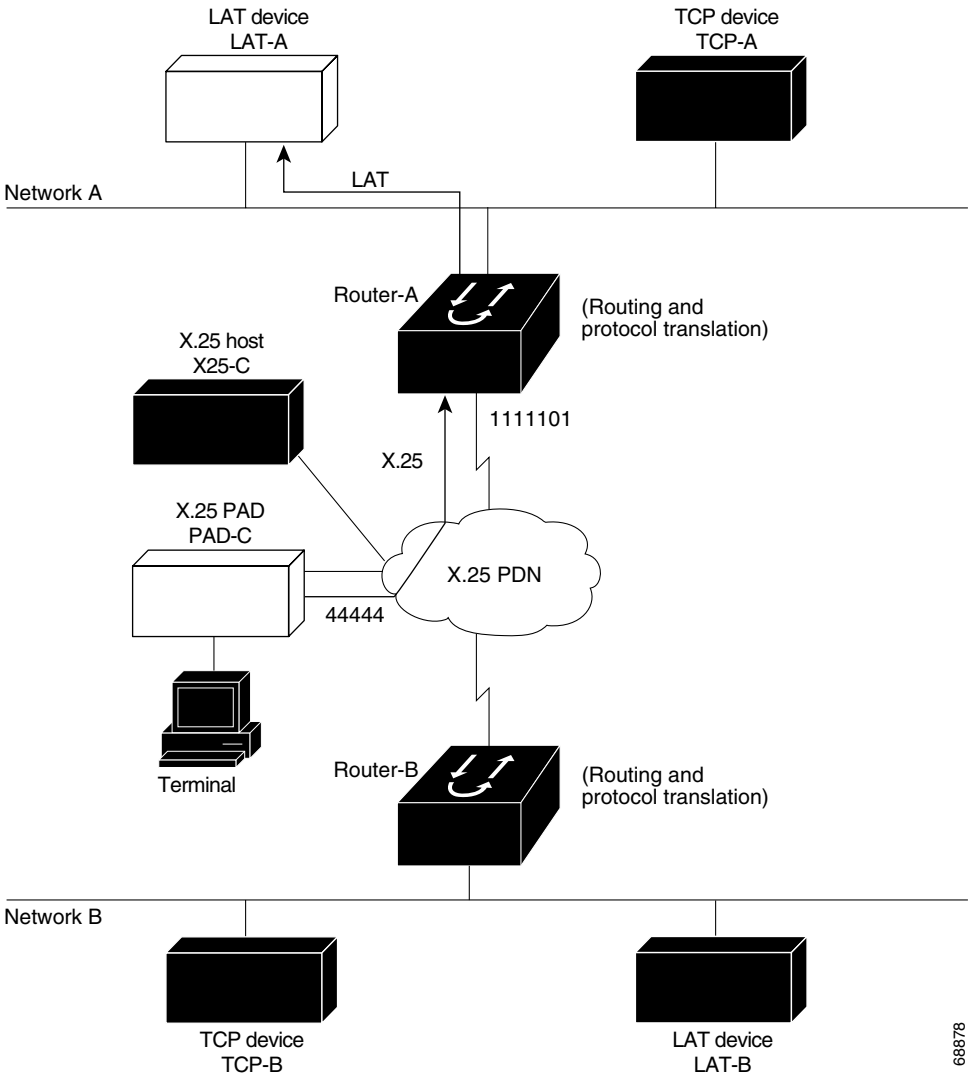
```
x29 profile linemode 2:1 3:2 15:1
translate tcp 172.16.1.26 x25 55551234 profile linemode
```

The X.3 PAD parameters are described in the “X.3 PAD Parameters” appendix at the end of this publication.

X.25 PAD-to-LAT Configuration Example

The following examples shows a protocol translation configuration that permits terminals connected to X.25 PADs to communicate with LAT devices on a remote LAN. (See Figure 22.) X.25 PAD terminals make a call using an X.121 address, which is translated to a LAT node. To the PAD terminal user, the connection appears to be a direct connection to a host on the X.25 PDN. The Cisco IOS software also supports X.29 access lists, which allow you to restrict LAN resources (LAT or TCP) available to the PAD user.

Figure 22 X.25 PAD-to-LAT Translation



The following example shows how to use the **translate** global configuration command to translate from an X.25 PAD to a LAT device on Network A. It is applied to Router-A. The configuration example includes an access list that limits remote LAT access through Router-A to connections from PAD-C.

```
! Define X25 access list to only allow pad-c.
x29 access-list 1 permit ^44444
x29 access-list 1 deny .*
!
! Set up translation.
translate x25 1111101 lat LAT-A access-class 1
```

This configuration example typifies the use of access lists in the Cisco IOS software. The first two lines define the scope of access-list 1. The first line specifies that access list 1 will permit all calls from X.121 address 44444. The caret symbol (^) specifies that the first number 4 is the beginning of the address number. Refer to the appendix "Regular Expressions" at the end of this publication for details concerning the use of special characters in defining X.121 addresses. The second line of the definition explicitly denies calls from any other number.

This access list is then applied to all incoming traffic on the serial port for Router-A (X.121 address 1111101) with the third configuration line in the example. However, it applies only to the **translate** command at the end of this example. This **translate** command specifies that incoming X.25 packets on the serial line (with address 1111101) are translated to LAT and sent to LAT-A if they pass the restrictions of the access list.

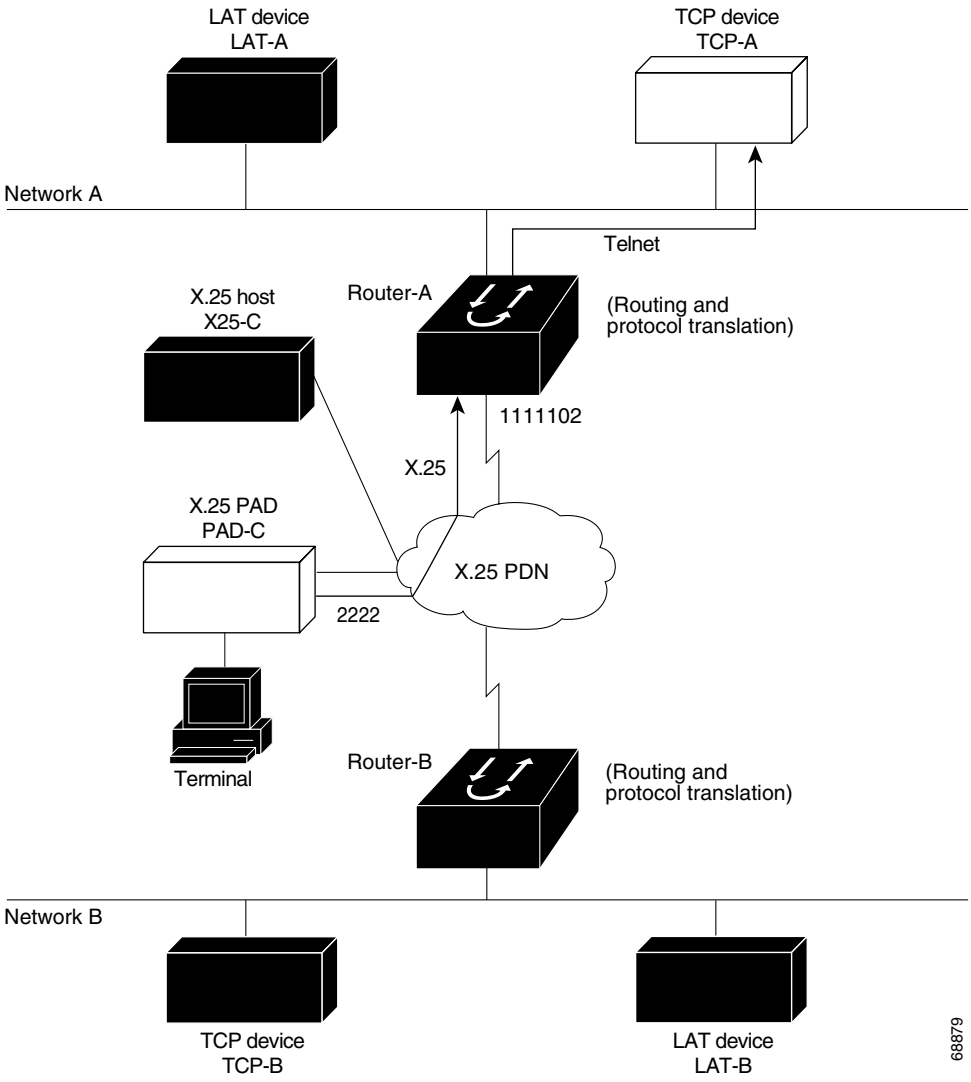
If you define multiple X.25 **translate** commands, each must contain a unique X.121 address. Also, the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) protocol that transfers packets must match the X.121 addresses. This requirement is specified in the protocol identification field of CUD. This field specifies whether a packet is routed, translated, or handled as a virtual terminal connection.

Note The X.121 address 1111101 used in this example can be a subaddress of the address 11111 originally assigned to this serial port on Router-A at the beginning of the configuration example section. However, making this assignment is not a requirement. The number to use in the **translate** command is negotiated (administratively) between your network management personnel and the PDN service provider. The X.121 address in the **translate** command represents the X.121 address of the calling device. That number may or may not be the number (or a subaddress of the number) administratively assigned to the router with the protocol translation option. You and the PDN must agree on a number to be used, because it is possible that the PDN can be configured to place calls that are intended for a destination on a given line that does not match the number assigned by you in the configuration file. Refer to the *1984 CCITT Red Book* specifications for more information concerning X.121 addresses.

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 23.) 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 23 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

The examples in the following sections show how to make connections for protocol translation using the one-step and two-step methods:

- One-Step Method for TCP-to-X.25 Host Connections Example
- Using the Two-Step Method for TCP-to-PAD Connections Example
- Two-Step Protocol Translation for TCP-to-PAD Connections Example

One-Step Method for TCP-to-X.25 Host Connections Example

This sample session 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 user name, 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. The 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, 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 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, you log out, which returns you to the router system EXEC prompt. From there, you enter the EXEC **quit** command, and the router drops the network connection to the PAD.

Two-Step Protocol Translation for TCP-to-PAD Connections Example

The following sample session 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 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, and 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, 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 the 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

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
```

Appendixes

Regular Expressions

This appendix explains regular expressions and how to use them in Cisco IOS software commands. It also provides details for composing regular expressions. This appendix has the following sections:

General Concepts About Regular Expressions

A regular expression is entered as part of a command and is a pattern made up of symbols, letters, and numbers that represent an input string for matching (or sometimes not matching). 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 uses regular expression pattern matching in several implementations. The following is a list of some of these implementations:

- BGP IP AS-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: b8 and W4, as examples

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. If a regular expression can match two different parts of an input string, it will match the earliest part first.

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 use 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. One and 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\-\]]

The previous 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, 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 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 that a pattern 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 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, you 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 and \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 use later.

Regular Expression Examples

This section provides the following practical examples of regular expression use:

- Regular Expression Pattern Matching in Access List Example
- Regular Expression Pattern Matching in Scripts Example
- Regular Expression Pattern Matching in X.25 Routing Entries Example
- Regular Expression Pattern Matching in a Protocol Translation Ruleset Example

Regular Expression Pattern Matching in Access List Example

Both the BGP IP autonomous system path feature and X.29 access list configuration statements can use regular expression patterns to match addresses for allowing or denying access.

The following BGP example contains the regular expression ^123.*. The 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
```

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 4.4.4.4 remote-as 4
 neighbor 4.4.4.4 route-map foo in

route-map foo permit 10
 match as-path 1
```

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
```

Regular Expression Pattern Matching in Scripts Example

On asynchronous lines, chat scripts send commands for modem dialing and logging in to remote systems. You use a regular expression in the **script dialer** command to specify the name of the chat script that the Cisco IOS software is to 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 the Cisco IOS software uses that chat script for the specified lines. For lines 1 and 6, the 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
```

Regular Expression Pattern Matching in X.25 Routing Entries Example

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 the Cisco IOS software is to remember for use later. The \1 in the rewrite pattern recalls the portion of the original address matched by the digits following the 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
```

Regular Expression Pattern Matching in a Protocol Translation Ruleset Example

Note Protocol translation rulesets are supported only in Cisco IOS Release 12.3(8)T and later software.

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.

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.