Protocol Translation Configuration Guide and Command Reference

Cisco Internetwork Operating System
Release 10.3

The products and specifications, configurations, and other technical information regarding the products contained in this manual are subject to change without notice. All statements, technical information, and recommendations contained in this manual are believed to be accurate and reliable but are presented without warranty of any kind, express or implied, and users must take full responsibility for their application of any products specified in this manual.

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual for this device, may cause interference to radio communications. This equipment has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference, in which case users at their own expense will be required to take whatever measures may be required to correct the interference.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Digital or an authorized sublicensor.

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

The following are trademarks of Digital Equipment Corporation: DDCMP, DEC, DECnet, DECNIS, DECserver, DECsystem, DECwindows, Digital, DNA, OpenVMS, ULTRIX, VAX, VAXstation, VMS, VMScluster, and the DIGITAL logo.

Portions of this document is used with permission of Cisco Systems, Incorporated. Copyright © 1990 - 1995, Cisco Systems, Inc.

The following third-party software may be included with your product and will be subject to the software license agreement:

CiscoWorks software and documentation are based in part on HP OpenView under license from the Hewlett-Packard Company. HP OpenView is a trademark of the Hewlett-Packard Company. Copyright © 1992, 1993 Hewlett-Packard Company.

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

Network Time Protocol (NTP). Copyright © 1992, David L. Mills. The University of Delaware makes no representations about the suitability of this software for any purpose.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

About This Manual

This section discusses the objectives, audience, organization, and conventions of the *Protocol Translation Configuration Guide and Command Reference*.

Document Objectives

This manual describes the tasks and the commands necessary to configure and maintain protocol translation in one or more of the following environments: Local Area Transport (LAT), TN3270, Serial Line Internet Protocol (SLIP) or Point-to-Point Protocol (PPP),XRemote, or TCP (Telnet). For information about routing X.25 and TCP/IP, refer to the *Router Products Configuration Guide* and the *Router Products Command Reference* publications and addenda.

Audience

This publication is intended for users who are responsible for configuring and maintaining a router running protocol translation. This publication provides the following levels of audience support:

- A task-oriented portion that includes task overviews, expanded descriptions of tasks, and comprehensive configuration examples for less-experienced users who need to understand the tasks as well as the commands.
- A command reference portion and appendixes for experienced users who just need reference information to complete a task. This portion describes tasks only in the context of using a particular command; it does not describe how the tasks interrelate or provide comprehensive configuration examples.

All users should have prior networking experience and should be familiar with their own network topologies.

Document Organization

This publication is divided into three main parts, each identified by a divider page, as follows:

- Part 1, “Configuring Protocol Translation”—An overview chapter and four chapters that provide task overviews, expanded descriptions of tasks, and comprehensive configuration examples.
- Part 2, “Protocol Translation Command Reference”—Four chapters that provide in-depth descriptions of the commands necessary for configuring and maintaining your router as a protocol translator. The chapters in Part 2 parallel the chapters in Part 1.
- Appendixes—Six appendixes that provide reference information related to the use of the commands.

For information on the following topics, refer to the applicable chapters in the *Router Products Configuration Guide* and the *Router Products Command Reference* publications and addenda:

- Understanding the user interface
- Loading system images, microcode images, and configuration files
- Configuring terminal sessions and modem support
- Managing the system
- Configuring interfaces
- Configuring Frame Relay
- Configuring Switched Multimegabit Data Service (SMDS)
- Configuring X.25
- Configuring TCP/IP

Document Conventions

This publication uses the following conventions:

- Ctrl and the symbol ^ represent the key labeled *Control*.

  For example, the combination *Ctrl-D* means hold down the *Control* key while you press the *D* key.

- A string is defined as a nonquoted set of characters.

  For example, when setting up a community string for SNMP to "public," do not use quotes around the string, or the string will include the quotation marks.

Command descriptions use these conventions:

- Vertical bars (|) separate alternative, mutually exclusive, elements.
- Square brackets ([ ]) indicate optional elements.
- Braces ({ }) indicate a required choice.
- Braces within square brackets ([{ }]) indicate a required choice within an optional element.
- **Boldface** indicates commands and keywords that are entered literally as shown.
- *Italics* indicate arguments for which you supply values; in contexts that do not allow italics, arguments are enclosed in angle brackets (< >).

Examples use these conventions:

- Examples that contain system prompts denote interactive sessions, indicating that the user enters commands at the prompt. The system prompt indicates the current command mode. For example, the prompt `CPT(config)#` indicates global configuration mode.
- Terminal sessions and information the system displays are in `screen` font.
- Information you enter is in **`boldface screen`** font.
- Nonprinting characters, such as passwords, are in angle brackets (< >).

- Default responses to system prompts are in square brackets ([ ]).
- Exclamation points (!) at the beginning of a line indicate a comment.

Note Means *reader take note*. Notes contain helpful suggestions or references to materials not contained in this manual.

CHAPTER 1

Product Overview

This section provides an overview of the products that support protocol translation. You will find the following information in this chapter:

- Protocol Translation Functionality
- Supported Transmission Protocols and Services
- Supported Interfaces and Connections
- Physical Configuration Options

Note In the context of this publication, a router set up to run protocol translation software is referred to as a router.

Protocol Translation Functionality

Routers are high-performance application-level gateways that can provide connectivity among systems running differing protocols and over a variety of media.

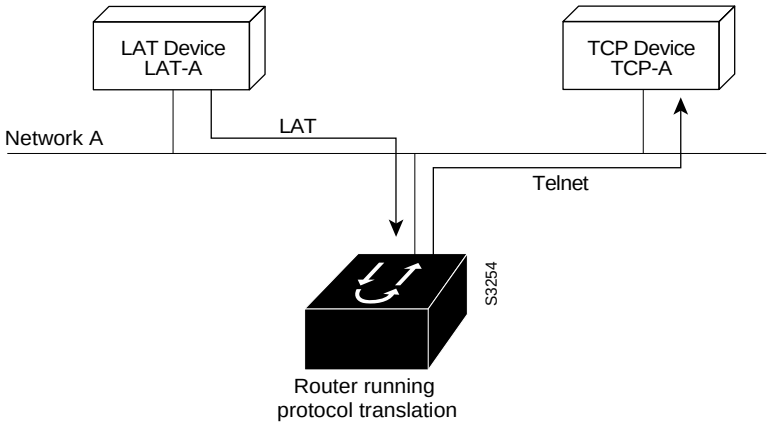
As part of their software capability, routers provide distributed network management facilities to assist in performance monitoring and run-time error logging, and support the Simple Network Management Protocol (SNMP). These facilities enable you to examine and adjust the routers for optimum performance.

Routers using protocol translation translate virtual terminal protocols, allowing devices running dissimilar protocols to communicate. The protocol translation software supports Telnet (called TCP for Transmission Control Protocol in the configuration syntax of protocol translation software), Local Area Transport (LAT), Serial Line Internet Protocol(SLIP) and Point-to-Point Protocol (PPP), and X.25. One-step protocol translation software performs bidirectional translation between any of the following protocols:

- Telnet and LAT
- Telnet and X.25
- LAT and X.25
- Telnet and SLIP/PPP
- LAT and SLIP/PPP
- X.25 and SLIP/PPP

Figure 1-1 illustrates LAT-to-Telnet protocol translation.

Figure 1-1 LAT-to-Telnet Protocol Translation



Note TN3270 and XRemote are also supported by the protocol translation software. However, to translate between these and other supported protocols, you must use the two-step method. For information about two-step translations in general, refer to the *Cisco Access Connection Guide*. Refer to the chapters "Configuring TN3270," "Configuring SLIP and PPP," and "Configuring XRemote" later in this publication for task-oriented configuration information about TN3270, SLIP and PPP, and XRemote. Refer to the chapters "TN3270 Configuration Commands," "SLIP and PPP Configuration Commands," and "XRemote Configuration Commands" later in this publication for detailed command descriptions for TN3270, SLIP and PPP, and XRemote.

Supported Transmission Protocols and Services

Routers provide a flexible set of capabilities for making connections using different media and between different hosts and resources running different protocols. The following descriptions summarize the protocols and connection services supported by routers:

- Transmission Control Protocol/Internet Protocol (TCP/IP)—The most widely implemented protocol suite on networks of all media types. TCP/IP is today's standard for internetworking and is supported by most computer vendors, including all UNIX-based workstation manufacturers.
- SLIP and PPP—Encapsulation methods that provide an inexpensive way of connecting PCs over asynchronous lines to a network. SLIP and PPP over asynchronous dial-up modems allow a home computer to be connected to a network without the cost of a leased line.
- Local Area Transport (LAT) protocol—Digital Equipment Corporations's proprietary terminal connection protocol used with Digital minicomputers. Router/bridges support bridging of the LAT protocol. Routers translate LAT packets to X.25, Telnet, and TN3270.
- X.25 PAD protocols—Cisco routers support the X.25 protocol and X.3/X.28/X.29 specifications.
- IBM 3278 terminal emulation—The Cisco implementation of TN3270 terminal emulation provides TN3270-based connectivity—specifically, emulation of IBM 3278-2 terminals—to IBM hosts over serial lines.

- Network Computing Devices Inc. XRemote terminal facility—XRemote is a protocol developed specifically to optimize support for X Window operation at a terminal over a serial communications link. XRemote allows for remote X Window operation using an NCD terminal.

Supported Interfaces and Connections

In addition to supporting Ethernet (the 802.3 specification of the Institute of Electrical and Electronic Engineers [IEEE]), routers support synchronous serial circuits at many speeds and can be connected to two serial lines. Router serial interfaces are can of transmit and receive data at up to four megabits per second, and support connectivity to WAN services such as Switched Multimegabit Data Service (SMDS), Frame Relay, and X.25.

A broad line of media adapters are also available for your convenience, including RS-232, V.35, X.21, and RS-449.

Physical Configuration Options

This section describes the router models that can be set up to run protocol translation software and the microprocessors these models use.

Platforms Supporting Protocol Translation

Internetwork Operating System (IOS) Release 10.3 supports protocol translation on the following router platforms:

- Cisco 4000 series (including the Cisco 4500)—A multiprotocol router that supports up to three network processor modules at a time, including Ethernet, Token Ring, and Dual Serial interfaces. The Cisco 4000 comes standard with Flash EPROM and is ideal for use in branch office or remote environments. Protocol translation is provided as a software option, thereby supporting concurrent routing, bridging, and protocol translation capabilities.
- Cisco 3000 series—A two-port multiprotocol router in a fixed configuration with Ethernet, Token Ring, serial, and ISDN BRI interface options. The ISDN-BRI interface is composed of two B channels and one D channel for circuit-switched communication of voice, data, and video. The Cisco 3000 comes standard with Flash memory and is ideal for use in branch office or remote environments. Protocol translation is provided as a software option, thereby supporting concurrent routing, bridging, and protocol translation capabilities.
- Cisco 2500 series—A two-port multiprotocol router and access server. As a router, it has a fixed configuration with Ethernet, Token Ring, synchronous serial, and ISDN BRI interface options. As an access server, it has a fixed configuration with Ethernet, synchronous serial, and either 8 or 16 asynchronous lines. A variety of protocol translation software feature sets permits the Cisco 2500 series router to be tailored to the needs of specific remote environments. A Cisco 2500 series router can also be configured to connect a local SDLC device and a local-area network (LAN) to a corporate internetwork using the second synchronous serial port.

Note Protocol translation is also supported on the 500-CS and the ASM-CS communication server platforms. Protocol translation on these platforms is described in the *Access and Communication Servers Configuration Guide*.

Microprocessors

The Cisco products listed in the previous section use either the MC68020, MC68030, or MC68040 microprocessor for high-speed operation. All microprocessors contain onboard RAM, Flash memory, system ROM holding all operating system, bootstrap, and diagnostic software, and hardware and software support for a control console.

Routers also provide nonvolatile memory that retains configuration information despite power losses or system reboots. With nonvolatile memory, the terminal and network servers do not need to rely on other network servers for configuration and boot service information.

PART 1

Configuring Protocol Translation

CHAPTER 2

Configuring LAT

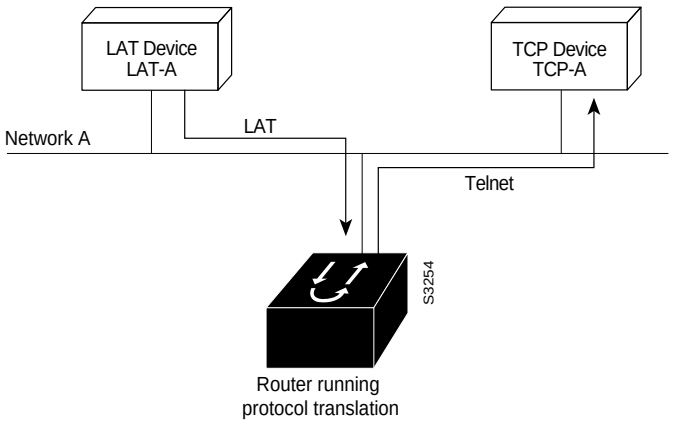
The Digital Equipment Corporation (Digital) Local Area Transport (LAT) protocol is the one used most often to connect to Digital hosts. LAT is a Digital-proprietary protocol. Cisco provides LAT technology licensed from Digital. This chapter describes how to configure the LAT transmission protocol on the routers. For a complete description of the commands in this chapter, see the chapter "LAT Configuration Commands" later in this publication. For information about establishing LAT connections, refer to the *Cisco Access Connection Guide*.

Cisco's Implementation of the LAT Protocol

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

Unlike the Transmission Control Protocol/Internet Protocol (TCP/IP), LAT was designed to be used on local-area networks (LANs) and it cannot be routed because it does not have a routing layer. However, a bridge or combined bridge and router, such as the Cisco router, can be used to carry LAT traffic across a wide-area network (WAN). Protocol translation can be used to carry LAT traffic over a WAN by first translating LAT to X.25 or Telnet, as shown in Figure 2-1.

Figure 2-1 Comparing LAT and TCP/IP Protocol Stacks



Cisco supports the LAT Version 5.2 specification.

Master and Slave Functionality

The LAT protocol is asymmetrical; it has master and slave functionality. First, the LAT master starts a LAT circuit by sending a circuit start message, and then a LAT slave responds with its own circuit start message. From 1 to 255 LAT sessions can then be multiplexed on this circuit.

In a typical setup, where a user's terminal is connected to a router, the router acts as the master, and the target VMS host acts as the slave. For example, the following command results in the router PT1 acting as the master (or server) and the target VMS host, wheel, acting as the slave (or host).

```
PT1> lat wheel
```

A router can also act as a slave. This happens if the user connects from one router to another. For example, the following command results in PT1 acting as the master (server) and PT2 acting as the slave (host).

```
PT1> lat pt2
```

In a LAT host-initiated connection, the VMS system always acts as the LAT slave. For example, a print job originating from a VMS system initiates or triggers the router to which the printer is connected to act as the LAT master. In short, the master-slave relationship also applies to host-initiated sessions from a LAT slave.

LAT Services

Resources such as modems, computers, and application software are viewed in a LAT network as services that, potentially, any user in the network can use. A LAT node can offer one or more such LAT services and more than one LAT node can offer the same LAT service.

A LAT node that offers one or more services, collectively called advertised services, broadcasts its services in the form of Ethernet multicast messages, called LAT service announcements. Conversely, a LAT node can listen for LAT service announcements on the network. These messages are cached in a dynamic table of known LAT services, collectively called learned services.

Your router supports both learned and advertised services and, therefore, it also supports incoming and outgoing LAT sessions. The services rating of its advertised nodes are determined dynamically but can also be set statically.

To establish outgoing connections to a LAT service, the router searches for the service in the learned services cache. If one or more nodes is offering the same service, the node with the highest rating is chosen. For example, a LAT connection to a service offered by a VAXcluster connects to the node in that cluster that has the smallest load and thus the highest service rating. This is how load-balancing works in relation to a group of nodes offering the same service. An incoming LAT session connects from another LAT node to the service advertised by the local LAT node.

LAT Groups

Because potentially any user on a LAT network can access any of the services on the network, a LAT server manager uses the concept of *group codes* to allow or restrict access to the services.

When the group codes on both the router and the LAT host share a common group code, a connection can be established between the two. If the default group codes have not been changed on either the routers or the LAT hosts, a user on any router can connect to any learned service on the network.

You might want to design a plan that correlates group numbers with organizational groups, such as departments. You can define subgroups using the line configuration commands described in this chapter. The section "Define a Group List for Outgoing Connections" later in this chapter describes how to enter group code lists in your router configuration file.

If you define groups for routers and LAT hosts, you can partition these services into logical subnetworks. You can organize the groups so that users on one router view one set of services, and users on another router (or another line on the same server) view a different set.

A LAT host node's services cannot be accessed individually; access is granted, per node, on an all-or-none basis.

LAT Sessions and Connection Support

A LAT session is a two-way logical connection between a LAT service and the router. The two-way connection is transparent to the user at a console connected to a LAT session; to the user it appears that connection has been made to the desired device or application program.

When a host print job connects to a router, this is called a *host-initiated connection*. The router maintains a queue of hosts requesting connection by sending periodic status messages to the requesting host.

A host-initiated connection is one in which the host connects to the router by specifying a port number or by defining a service on the router, such as a print job. These same services are used for connections from other routers.

Note If a connection request is received specifying a service and a destination port name, the port name is used to determine the line number for connection purposes. This capability allows a user to connect to a specified port of the router simply by specifying any service on the server and a port number. (Earlier versions of the protocol translation software ignored the service name on inbound connections.)

Connecting a VMS Host to a Router Using LAT

Connecting a VMS host to a router using LAT is slightly different if you are connecting to a VMS host running VMS Version 5.4 or earlier or VMS Version 5.5 or later software.

VMS Version 5.4 or Earlier

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

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

VMS Version 5.5 or Later

Turn on the VMS LAT host's outgoing connections and use the Digital **set host/lat** command, as shown in the following example:

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

Port Names When Configuring a LAT Printer

When you configure a LAT printer, the LAT port name is the same as the line number, but without the "tty." For example, if you configure line tty10 of your router, named *ABLE*, to be a LAT printer port, the OpenVMS command to associate an arbitrary LTA device to a LAT port name is as follows:

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

The LAT port name is the line number without the "tty," regardless of whether the format of the tty line number is decimal or octal. Refer to the *Router Products Configuration Guide* for more information about configuring the router's line to the correct terminal characteristics (for example, baud rate, EXEC, flow control, and so forth).

Additional LAT Capability

Cisco's protocol translation software fully supports the LAT protocol suite, and provides the following features:

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

LAT Configuration Task List

Cisco's LAT protocol is supplied with a default configuration and does not require additional configuration for you to use it. The software does provide commands for customizing the LAT software for your environment, if desired.

Perform the tasks in the following sections to enable LAT, to customize LAT for your particular network environment, and to monitor and maintain LAT connections:

- Enable LAT
- Define a LAT Node Name
- Define a Group List for Outgoing Connections
- Specify a Group List Logical Name
- Specify Groups to Be Advertised
- Enable Inbound Services
- Control Service Announcements and Service Solicitation
- Configure Traffic Timers
- Optimize Performance
- Define Access Lists
- Monitor and Maintain LAT

See the section "LAT Examples" at the end of this chapter for ideas on how to configure LAT in your network.

Enable LAT

LAT is enabled by default on the router. If it has been disabled, you must reenable it. Perform the following tasks in interface configuration mode:

Task	Command
Enable the LAT protocol.	**lat enabled**
Disable the LAT protocol.	**no lat enabled**

Define a LAT Node Name

You can give the router a node name that is different than the host name. Perform the following task in global configuration mode:

Task	Command
Define a LAT node name.	**lat node** *node-name*

Define a Group List for Outgoing Connections

You can define the list of services to which a user can connect. Do this by defining the group code lists used for connections from specific lines. Perform the following task in line configuration mode:

Task	Command
Define the group list for an outgoing connection on a specified line.	**lat out-group** {*groupname* | *number* | *range* | **all**}

You can limit the connection choices for an individual line by defining the group code lists for an outgoing connection. When a user initiates a connection with a LAT host, the user's line must share a common group number with the remote LAT host before a connection can be made.

Specify a Group List Logical Name

You can specify a name for group lists to simplify the task of entering individual group codes. A name makes it easier to refer to a long list of group code numbers. Perform the following task in global configuration mode:

Task	Command
Specify logical names for group lists.	**lat group-list** *groupname* {*number* | *range* | **all**} [**enabled** | **disabled**]

To display the defined groups, use the **show lat groups** command.

Specify Groups to Be Advertised

You can specify a group code mask to use when advertising all services for a node. You can enter more than one group code by listing the numbers. You can also enter both a group code name and group codes. Perform the following task in global configuration mode:

Task	Command
Specify logical names for group lists.	**lat service-group** {*groupname* | *number* | *range* | **all**} [**enabled** | **disabled**]

Enable Inbound Services

Just as LAT services are offered by host computers, they can also be offered by routers. A router implements both the host and server portions of the LAT protocol. This allows connections from either hosts or routers. When a host connects to a router, this is called a *host-initiated connection*. Host-initiated connections and connections from other routers are referred to collectively as *inbound connections*.

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

Task	Command
Set the LAT password for a service.	**lat service** *service-name* **password** *password*
Set the LAT service ID for a specific service.	**lat service** *service-name* **ident** *identification*

Task	Command
Specify a static service rating for a specific service.	**lat service** *service-name* **rating** *static-rating*
Configure a LAT rotary group.	**lat service** *service-name* **rotary** *group*
Associate a command with a specific service for automatic execution.	**lat service** *service-name* **autocommand** *command*
Enable inbound connections to a specific service.	**lat service** *service-name* **enabled**

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

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

Control Service Announcements and Service Solicitation

You can configure your router to support the service responder feature that is part of the LAT Version 5.2 specification.

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

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

Your router can be configured as a LAT service responder. Of course, if all your nodes are LAT Version 5.2 nodes, you do not need to enable the service responder features.

You can control service announcements and service solicitations as described in the following sections:

- Enable Proxy Solicitation Support
- Disable Broadcasts of Service Announcements
- Adjust the Time between Service Announcements

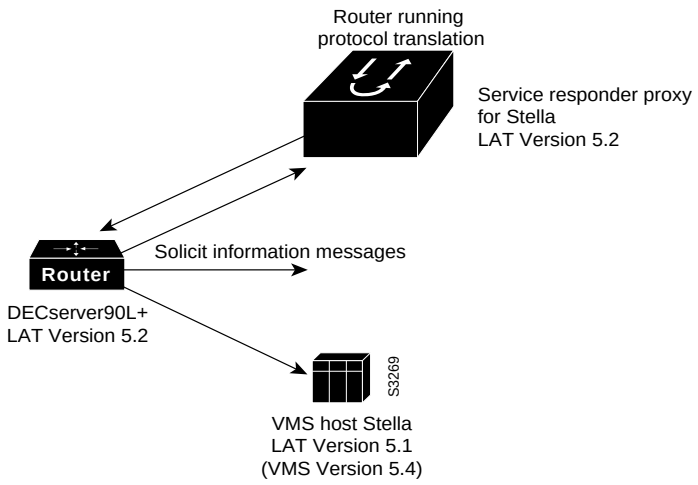
Enable Proxy Solicitation Support

You can configure the router to respond to solicit-information requests addressed to LAT Version 5.1 nodes. This function allows nodes that do not cache service announcemets to interoperate with nodes that do not respond to solicit requests. Perform the following task in global configuration mode:

Task	Command
Enable a proxy node to respond to solicit-information multicast messages.	**lat service-responder**

Figure 2-2 shows how a router can act as a proxy for LAT servers.

Figure 2-2 Router as Proxy for LAT Server



The DECserver90L+ broadcasts a solicit-information request in search of service *stella*'s address. The VMS host, Stella, is unable to respond to the request because it is running LAT Version 5.1. The router is running LAT Version 5.2 with theservice responder feature enabled and informs the DECserver90L+ of Stella's address.

Disable Broadcasts of Service Announcements

You can disable periodic broadcasts of service announcements. If service announcements are enabled, the LAT node will periodically broadcast service announcements. If service announcements are disabled, the LAT node will not send service announcements; therefore, a remote node requiring a connection to such a node must use solicit-information messages to look up node information. Perform the following task in global configuration mode:

Task	Command
Disable periodic broadcasts of service announcements.	**no lat service-announcements**

Only disable service announcements if all of the nodes on the LAN support the service responder feature.

Adjust the Time between Service Announcements

You can adjust the time between LAT service announcements for services offered by the router. This is useful in large networks with many LAT services and limited bandwidth. Perform the following task in global configuration mode:

Task	Command
Adjust the time between service announcements.	**lat service-timer** *interval*

Configure Traffic Timers

You can customize the environment for transmitting LAT messages. Our implementation of LAT allows you to set these features:

- The number of retransmissions before declaring a system unreachable (the message-retransmit limit)
- The interval of time LAT waits before sending a keepalive message on an idle connection (the keepalive timer)
- The interval of time LAT waits between transmission of messages (the virtual-ciruit timer)

These features affect all LAT connection types. Perform the following tasks in global configuration mode, as necessary:

Task	Command
Set the message-retransmit limit.	**lat retransmit-limit** *number*
Set the keepalive timer.	**lat ka-timer** *seconds*
Set the virtual-circuit timer.	**lat vc-timer** *milliseconds*

Optimize Performance

You can optimize performance for your LAT environment by performing the tasks described in the following sections:

- Set the Number of Sessions on a Virtual Circuit
- Set the Number of Messages Received by a Host Node
- Set the Number of Messages Received by a Server Node

Set the Number of Sessions on a Virtual Circuit

You can set the number of sessions multiplexed over a single a LAT virtual circuit. The maximum (and default) number of sessions is 255. Perform the following task in global configuration mode:

Task	Command
Set the maximum number of sessions on a LAT virtual circuit.	**lat vc-sessions** *number*

Set the Number of Messages Received by a Host Node

You can set the number of messages received by a host at one time. Increasing this number can enhance performance. Before Version 5.2, LAT allowed only one outstanding message at a time on a virtual-circuit. This restriction could limit the performance of routers processing a large number of messages because only one Ethernet packet of data could be in transit at a time. With LAT Version 5.2, nodes can indicate that they are willing to receive more than one message at a time. During virtual circuit startup, each side communicates to the other how many outstanding messages it is willing to accept. Perform the following task in global configuration mode:

Task	Command
Allow LAT host node to receive more than one message at a time.	**lat host-buffers** *receive-buffers*

Set the Number of Messages Received by a Server Node

You can set the number of messages received by a server at one time. Increasing this number can enhance performance. Before Version 5.2, LAT allowed only one outstanding message on a virtual circuit at a time. This restriction could limit the performance of routers processing a large number of messages because only one Ethernet packet of data could be in transit at a time. With LAT Version 5.2, nodes can indicate that they are willing to receive more than one message at a time. During virtual-circuit startup, each side communicates to the other how many outstanding messages it is willing to accept. Perform the following task in global configuration mode:

Task	Command
Allow a LAT server node to receive more than one message at a time.	**lat server-buffers** *receive-buffers*

Define Access Lists

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

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

To define access lists and conditions, perform the following task in global configuration mode:

Task	Command
Specify an access condition.	**lat access-list** *number* {**permit** | **deny**} *nodename*

To restrict connections between terminal lines and node names in an access list, perform the following task in line configuration mode:

Task	Command
Restrict incoming and outgoing connections between a particular terminal line or group of lines and the node names in an access list.	**access-class** *access-list-number* {**in** | **out**}

Monitor and Maintain LAT

To monitor and maintain LAT activity, perform one or more of the following tasks in EXEC mode:

Task	Command
Delete an entry from the queue.	**clear entry** *number*
Display queued host-initiated connections.	**show entry**
Display LAT services offered to other LAT systems.	**show lat advertised**
Display defined LAT groups.	**show lat groups**
Display information about LAT nodes.	**show lat nodes**
Display active LAT sessions.	**show lat sessions** [*line-number*]

Task	Command
Display traffic and resource utilization statistics.	**show lat traffic**
Display information about LAT nodes. (Information is displayed in the same way as in the Digital interface.)	**show node** [**all** | *node-name*] [**counters** | **status** | **summary**]
Display LAT learned services.	**show service** [*service-name*]

LAT Examples

The following sections provide LAT configuration examples:

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

Establishing Basic LAT Service Example

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

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

Establishing a LAT Service with Selected Group Codes Example

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

```
! Establish a LAT group list
lat group-list HUBS 1 4-7 167
!
! Enable LAT group list for the service-group
lat service-group HUBS enabled
!
! Create LAT service with password protection and
! identification string
lat service ABLE password secret
lat service ABLE ident Welcome to my machine
```

Displaying the LAT Services on the Same LAN Example

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

```
able> show lat services

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

Establishing an Outbound LAT Session Example

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

```
able> lat HELLO
```

Logically Partitioning LAT Services by the Terminal Line Example

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

```
! Define LAT groupnames
lat group-list DEFAULT 0
lat group-list FLOOR 3
lat group-list QA 4
!
line 1 7
lat out-group FLOOR enabled
lat out-group DEFAULT disabled
line 8 11
lat out-group QA enabled
lat out-group DEFAULT disabled
line 12 16
lat out-group DEFAULT FLOOR QA enabled
```

Configuring LAT Rotary Groups Example

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

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

LAT Access Lists Examples

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

```
! Permit all IP hosts, LAT nodes beginning with “VMS” and no X.25
! connections to the printer on line 5
!
access-list 1 permit 0.0.0.0 255.255.255.255
lat access-list 1 permit ^VMS.*
x29 access-list 1 deny .*
!
line 5
access-class 1 in
!
! Meanwhile, permit outgoing connections to various places on all the
! other lines.
!
! Permit IP access within cisco
access-list 2 permit 131.108.0.0 0.0.255.255
!
! Permit LAT access to the Stella/blue complexes.
lat access-list 2 permit ^STELLA$
lat access-list 2 permit ^BLUE$
!
! Permit X25 connections to infonet hosts only.
x29 access-list 2 permit ^31370
!
line 0 99
access-class 2 out
```

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

```
access-list 1 permit 131.108.0.0 0.0.255.255
access-list 1 permit 150.136.0.0 0.0.255.255
!
line 1 40
access-class 1 out
! define LAT access list that permits all connections
lat access-list 1 permit .*
```

Associating a Rotary Group with a Service Example

The following example defines a service that communicates with a specific line and defines a rotary with only that line specified. For more information about rotary groups and the **rotary** line configuration command, refer to the *Router Products Configuration Guide*.

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

```
rotary 2
lat service ciscopt2 rotary 2
lat service ciscopt2 enable
```

CHAPTER 3

Configuring TN3270

IBM 3270 display terminals are among the computing community's most widely implemented and emulated environments for host-based computing. This chapter will help you understand the TN3270 terminal emulation environment and how to use and create files that allow terminals connected to the routers to be used for TN3270 operation. For a complete description of the commands in this chapter, refer to the chapter "TN3270 Configuration Commands" later in this publication. For information about establishing TN3270 connections, refer to the *Cisco Access Connection Guide*.

Cisco's Implementation of TN3270

The TN3270 terminal emulation software is based on software developed at the University of California, Berkeley. This software allows any terminal to be used as an IBM 3270-type terminal. Users with non-3270 terminals can take advantage of the emulation capabilities to perform the functions of an IBM 3270-type terminal. Specifically, Cisco's implementation supports emulation of an IBM 3278-2 terminal providing an 80-by-24 display.

True IBM 3270-type terminals use a character format referred to as extended binary-coded decimal interchange code (EBCDIC). EBCDIC consists of 8-bit coded characters and was originally developed by IBM. Emulation is made possible by *termcap* and *curses* functions developed by Berkeley UNIX system developers. These functions translate the keyboard and terminal characteristics for ASCII-type terminals into those expected by an IBM host. ASCII characters are listed in the appendix "ASCII Character Set" later in this publication.

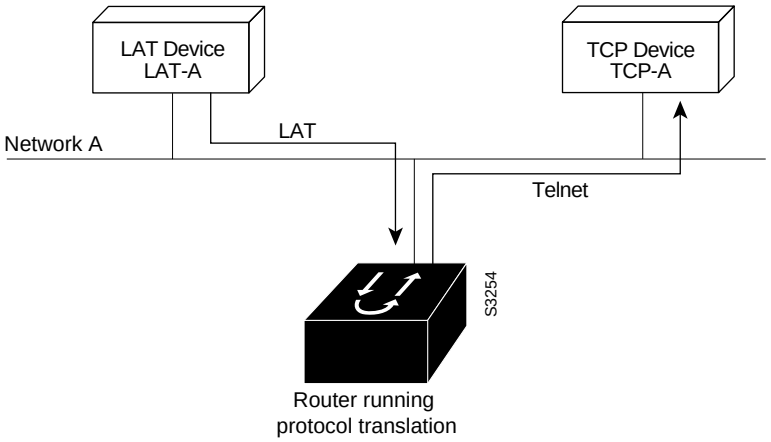
Formally, a termcap is a two-part terminal-handling mechanism. It consists of a database and a subroutine library. The database describes the capabilities of each terminal to be supported; the subroutine library allows programs to query the database and to make use of the values it contains. For more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

Routers include a default termcap entry for Digital VT100 terminal emulation. More samples are available directly from Cisco on the *ftp.cisco.com* directory using the File Transfer Protocol (FTP) utility.

TN3270 emulation capability allows users to access an IBM host without using a special IBM server or a UNIX host acting as a server (see Figure 3-1). The IBM host must directly support TCP/IP, or have a front-end processor that supports TCP/IP.

Connection to IBM hosts from LAT, TCP, and X.25/PAD environments is accomplished using the two-step translation method. This method is discussed in the chapter "Configuring Protocol Translation Sessions" in this publication and in the *Cisco Access Connection Guide*. In general, TN3270 support for protocol translation allows outgoing TN3270 connections only. In other words, LAT, TCP, and X.25/PAD users must first establish a connection with the router, then use the TN3270 facility from the router to make a connection to the IBM host.

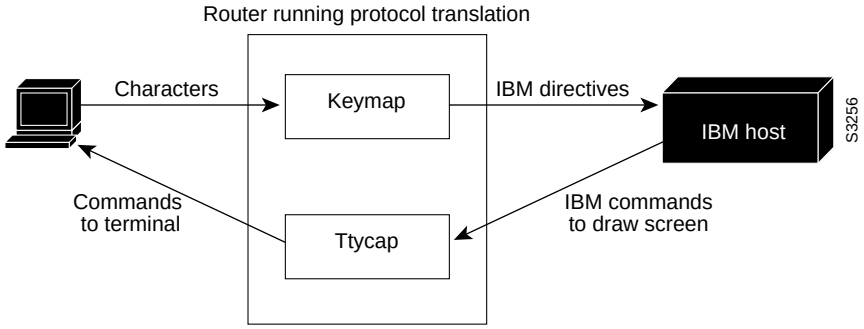
Figure 3-1 Typical 3270 Connection Environment



Keymaps and Ttycaps

Figure 3-2 shows how the keymapping and ttycap functionality on the router helps IBM hosts and non-IBM terminals communicate.

Figure 3-2 Keymaps and Ttycaps



Keymaps and ttycaps have the following functionality:

- Keymapping—Terminals send a key sequence for every key used to send packets to an IBM host.The keymapping function on the router identifies special sequences and converts them to directives to the IBM host. A minimal level of keymapping is supported by default. Several keys can convert to the same IBM directives.
- Ttycap—IBM sends commands to the terminal, including cursor position, clear screen, and so forth. The ttycap functionality on the router changes IBM directives into the terminal language. By default, routers conform to the ANSI terminal standard, which is VT*xxx* terminal compatible.

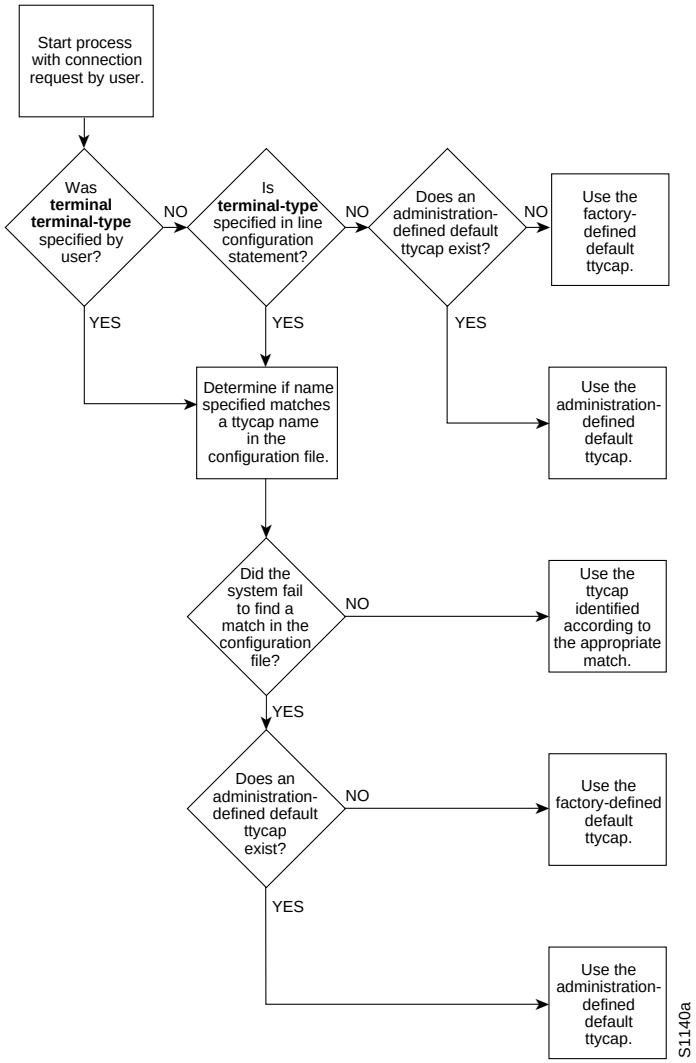
Startup Sequence Priorities

At system startup, the router uses the following decision sequence when selecting a terminal emulation file, also called a *ttycap*:

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

Figure 3-3 illustrates the decision process used by the router to choose a ttycap for a specific TN3270 session.

Figure 3-3 Typical TN3270 Ttycap Selection Process



At system startup, the router uses the following decision sequence when selecting a keyboard map file, also called a *keymap*:

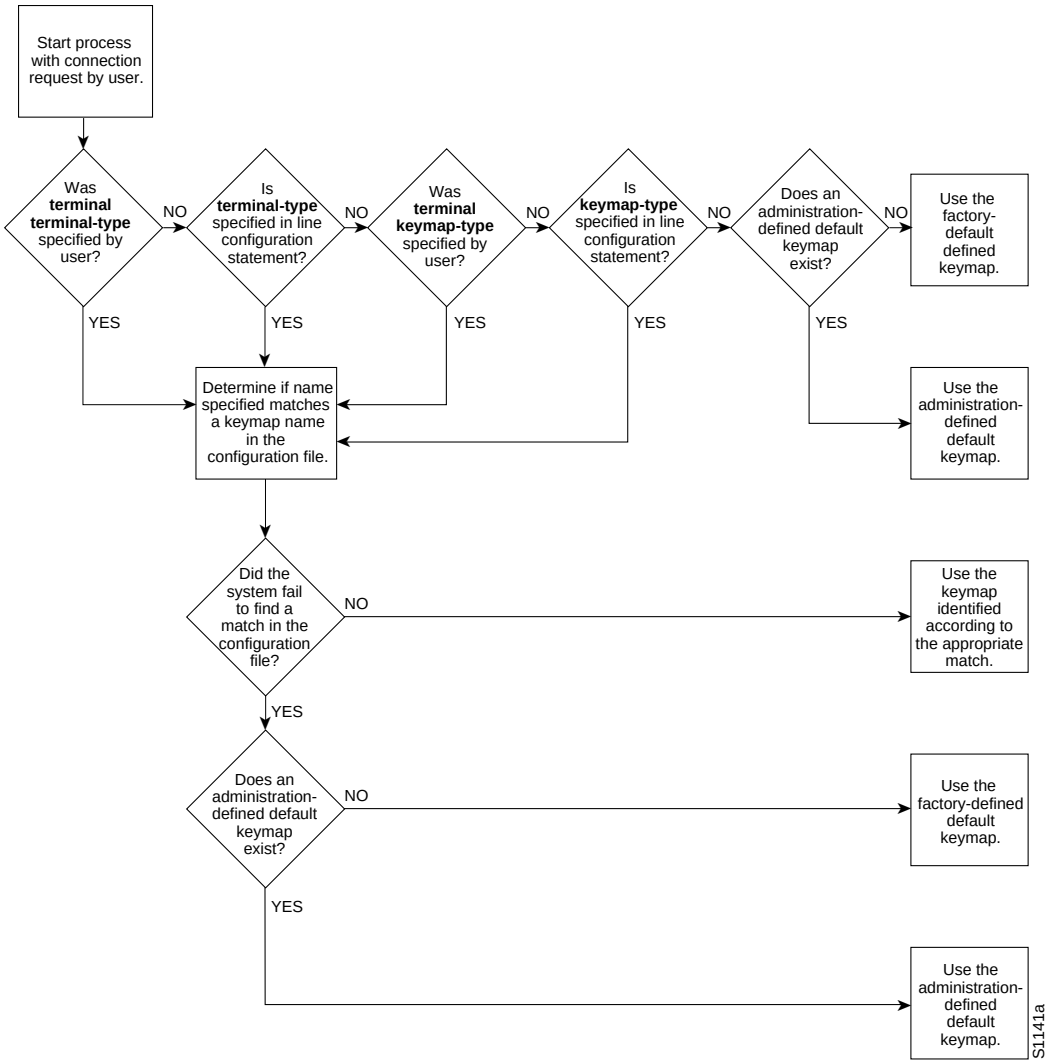
1. Use a user-supplied keyboard map filename.
2. Use a keyboard map filename specified using line configuration commands.
3. Use a user-supplied terminal emulation filename.
4. Use a terminal emulation filename specified using line configuration commands
5. Use the default keyboard map filename supplied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

The router uses the following criteria to determine which file to use:

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

Figure 3-4 illustrates the decision process used by the router to choose a keymap for a specific TN3270 session. When one of the first four priority checks fail (that is, the name specified does not match any name in the configuration file), the same rules listed for the terminal emulation file apply.

Figure 3-4 Typical TN3270 Keymap Selection Process



TN3270 Connection and Configuration Task List

You can perform the tasks described in the following sections when making connections to an IBM host:

- Use the Default VT100 Terminal and Keyboard Emulation
- Copy a Sample Terminal Emulation File
- Create a Custom Terminal Emulation File
- Assign Ttycap and Keymap Line Characteristics
- List the Terminal Emulation Files
- Map TN3270 Characters

See the section "TN3270 Configuration Examples" for examples of custom terminal and keyboard emulation files.

Use the Default VT100 Terminal and Keyboard Emulation

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

Enter the **tn3270** EXEC command at the EXEC prompt to connect to an IBM host. This command makes the connection using the terminal emulation file selected and using the startup sequence priorities outlined in the section "Startup Sequence Priorities" earlier in this chapter. For more information about the **tn3270** command, refer to the *Cisco Access Connection Guide.*

Copy a Sample Terminal Emulation File

If the default file does not work for your terminal and keyboard type or the host that you connect to, you might be able to find a file that will work from the growing list of sample terminal emulation files created by Cisco engineers and customers. Obtain the *tn3270.examples* file in the *ftp.cisco.com* directory at Cisco Systems. Numerous emulation files are listed in *tn3270.examples* that allow various terminal types to emulate an IBM 3270-type terminal.

The following steps describe how to obtain a sample configuration file from the *tn3270.examples* file available from the Cisco *ftp.cisco.com* directory. These steps assume that the system configuration file is stored on a host.

Step 1 Use the FTP utility to connect to address *ftp.cisco.com* and log in using the anonymous FTP convention.

Step 2 Get the *tn3270.examples* file.

Step 3 Use a text editor or word processing application to copy the sample terminal emulation file into the configuration file.

Step 4 Load the configuration file onto the host or network. (Refer to the chapter about loading system images, microcode images, and configuration files in the *Router Products Configuration Guide* for more information on loading configuration files.)

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

Create a Custom Terminal Emulation File

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

- You can provide default settings for all terminals in the network or terminals on a specific host.
- You can set up your system to boot, or load, a specific configuration file using configuration commands described in the chapter about loading system images, microcode images, and configuration files in the *Router Products Configuration Guide*.
- You can temporarily override default settings using terminal EXEC commands.
- You can use the local **terminal terminal-type** and **terminal keyboard-type** EXEC commands described in the *Cisco Access Connection Guide* to load in the files.
- You can configure line-specific emulation types for terminal negotiations with a remote host.

To create a custom terminal emulation file or a custom keyboard emulation file, perform the applicable global configuration task as follows:

Task	Command
Define a new terminal emulation file, or ttycap.	**ttycap** *ttycap-name termcap-entry*
Define a new keyboard emulation file, or keymap.	**keymap** *keymap-name keymap-entry*

Assign Ttycap and Keymap Line Characteristics

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

- Terminal type
- Keymap type

This information is used by the router when negotiating connections with hosts.

To assign ttycap and keymap line characters, perform one or more of the following line configuration tasks:

Task	Command
Specify the type of terminal connected to the line.	**terminal-type** *terminal-name*
Specify the keyboard map for a terminal connected to the line.	**keymap-type** *keymap-name*

List the Terminal Emulation Files

To display a list of ttycap and keymap files available for use, perform the following tasks in EXEC mode:

Task	Command
List the ttycap files available.	**show ttycap** [*ttycap-name* | **all**]
List the keymap files available.	**show keymap** [*keymap-name* | **all**]

Map TN3270 Characters

You can control the mapping of extended binary-coded decimal interface code (EBCDIC) and American Standard Code for Information Interchange (ASCII) characters by performing the tasks described in the following sections:

- Create Character Mappings
- Display Character Mappings
- Obtain the Hexadecimal Value
- Set Data Character Bits
- Allow the Use of Extended Features
- Specify a Null-Processing Option
- Specify Reset after Input Error

Create Character Mappings

You can create character mappings by configuring a two-way binding between EBCDIC and ASCII characters. To set character mappings, perform the following global configuration task:

Task	Command
Create a two-way binding between EBCDIC and ASCII characters.	**tn3270 character-map** *ebcdic-in-hex ascii-in-hex*

To return character mappings to their default settings, perform the following global configuration task:

Task	Command
Reset character mappings to defaults.	**no tn3270 character-map** {**all** | *ebcdic-in-hex*} [*ascii-in-hex*]

Display Character Mappings

To display character mappings, perform the following task in EXEC mode:

Task	Command
Display character mappings.	**show tn3270 character-map** {**all** | *ebcdic-in-hex*}

Obtain the Hexadecimal Value

To display the hexadecimal value of an ASCII character, perform the following task in EXEC mode:

Task	Command
Obtain the hexadecimal value of an ASCII character.	**show tn3270 ascii-hexval**

After you enter this command, type the ASCII character whose hexadecimal value you want to display.

Set Data Character Bits

When you create character mappings between EBCDIC or extended ASCII characters, you must configure the router for the correct data character bit length. The default mask used for TN3270 connections is a 7-bit mask. In certain situations, you must use an 8-bit display. When an 8-bit mask has been set, you can temporarily configure the router to use the 8-bit mask by performing the following task in line configuration command mode:

Task	Command
Temporarily configure the router to use the 8-bit mask.	**tn3270 8bit display**

When you use a file-transfer protocol such as Kermit in 8-bit mode or you use 8-bit graphics, which rely on transparent mode, perform the following line configuration task to configure the router for the 8-bit mask:

Task	Command
Configure the router to use the 8-bit mask.	**tn3270 8bit transparent-mode**

Allow the Use of Extended Features

You can allow the use of extended features by directing the router to append "-E" to the terminal type string sent to the IBM host.

To allow the use of extended features, perform the following line configuration task:

Task	Command
Append "-E" to the terminal type string sent to the IBM host, and allow the use of extended features.	**tn3270 datastream** {**extended** | **normal**}

Specify a Null-Processing Option

If a user enters data at a terminal keyboard, then uses an arrow key to move the cursor across the display screen and enters more data, the blanks spaces are filled with nulls. You can specify a method for processing null characters generated from user input at a terminal, emulating either a 3278x terminal in which nulls are compressed out of the entry string, or a 7171 controller in which nulls are replaced by spaces.

In line configuration mode, perform the following tasks to set a null-processing method and return null-processing to the default setting.

Task	Command
Specify the compression of nulls out of the user-entered string.	**tn3270 null-processing 3270**
Specify the conversion of nulls into spaces.	**no 3270 null-processing improved**

Specify Reset after Input Error

To specify that the terminal keyboard be locked until the user presses the Reset key, perform the following task in line configuration mode:

Task	Command
Lock the keyboard upon input error and do not permit any input until the user presses the Reset key.	**tn3270 reset-required**

TN3270 Configuration Examples

The following sections provide examples that will help you define custom terminal and keyboard emulation files and will help you configure your system to use these files:

- Custom Terminal Emulation File Example
- Custom Keyboard Emulation File Example
- Line Specification for a Custom Emulation Example
- Character Mapping Examples

Custom Terminal Emulation File Example

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

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

Custom Keyboard Emulation File Example

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

```
keymap ibm7171 \
vt100av|vt100|vt100nam|pt100|vt102|vt125{ \
enter = '^m';\
erase = '^?'; reset = '^g'; clear = '^z' | '\EOM';\
nl = '^j'; tab = '^i'; btab = '^b';\
left = '\EOD'; right = '\EOC'; up = '\EOA'; down = '\EOB';\
home = '^h'; delete = '^d'; eeof = '^e' | '\E^?'; einp = '^w'; insrt = '\EOn';\
pfk1 = '\EOP' | '\E1'; pfk2 = '\EOQ' | '\E2'; pfk3 = '\EOR' | '\E3';\
pfk4 = '\EOw' | '\E4'; pfk5 = '\EOx' | '\E5'; pfk6 = '\EOy' | '\E6';\
pfk7 = '\EOt' | '\E7'; pfk8 = '\EOu' | '\E8'; pfk9 = '\EOv' | '\E9';\
pfk10 = '\EOq' | '\E0'; pfk11 = '\EOr' | '\E-';\
```

```
pfk12 = '\EOs' | '\E='; pfk13 = '\EOp\EOP' | '^f13';\
pfk14 = '\EOp\EOQ' | '^f14'; pfk15 = '\EOp\EOR' | '^f15';\
pfk16 = '\EOp\EOw' | '^f16'; pfk17 = '\EOp\EOx' | '^f17';\
pfk18 = '\EOp\EOy' | '^f18'; pfk19 = '\EOp\EOt' | '^f19';\
pfk20 = '\EOp\EOu' | '^f20'; pfk21 = '\EOp\EOv' | '^f21';\
pfk22 = '\EOp\EOq' | '^f22'; pfk23 = '\EOp\EOr' | '^f23';\
pfk24 = '\EOp\EOs' | '^f24';\
pa1 = '^p1' | '\EOS';\
pa2 = '^p2' | '\EOm';\
pa3 = '^p3' | '\EOl';\
}
```

Line Specification for a Custom Emulation Example

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

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

Character Mapping Examples

The following example shows the configuration of the EBCDIC, ASCII character mappings listed in Table 3-1.

Table 3-1 Sample EBCDIC, ASCII Character Mapping

EBCDIC	ASCII
a	x
b	y
c	z

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

The following example displays all nonstandard character mappings:

```
PT# show tn3270 character-map all

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

The following example shows the standard key mapping for *d*:

```
PT# show tn3270 character-map 83

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

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

```
PT# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
PT(config)# no tn3270 character-map 0x80 0x78
PT(config)# ^Z
PT# show tn3270 character-map all
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
PT# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
PT(config)# no tn3270 character-map 0x82
PT(config)# ^Z
PT# show t3270 character-map all
EBCDIC 0x82 <=> 0x79 ASCII
```

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

```
PT# show tn3270 character-map all
EBCDIC 0x81 <=> 0x78 ASCII
EBCDIC 0x82 <=> 0x79 ASCII
EBCDIC 0x83 <=> 0x7A ASCII
PT# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
PT(config)# no tn3270 character-map all
PT(config)# ^Z
PT# show tn3270 character-map all
```

CHAPTER 4

Configuring SLIP and PPP

This chapter describes how to configure asynchronous interfaces for telecommuting applications using Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP) encapsulation. See the chapter "SLIP and PPP Configuration Commands" later in this publication for a complete description of the commands mentioned in this chapter.

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

Cisco's Implementation of SLIP and PPP

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

Using SLIP or PPP encapsulation over asynchronous lines is an inexpensive way of connecting PCs to a network. SLIP and PPP over asynchronous dial-up modems allow a home computer to be connected to a network without the cost of a leased line. Dial-up SLIP and PPP links can also be used for remote sites that need only occasional telecommuting or backup connectivity. Both public-domain and vendor-supported SLIP and PPP implementations are available for a variety of computer applications.

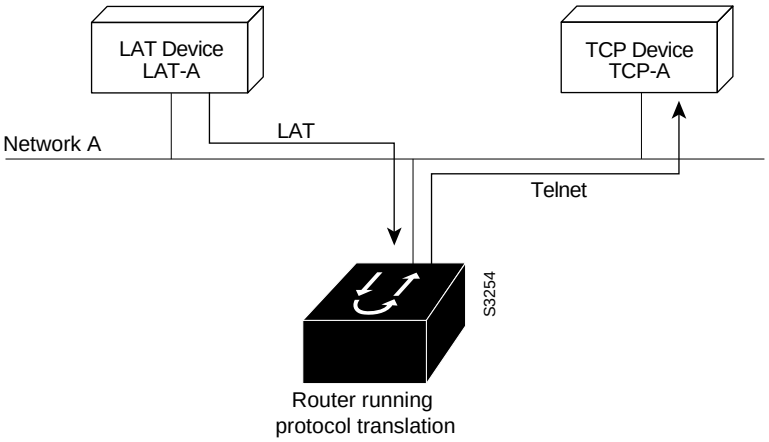
The router concentrates a large number of SLIP or PPP PC or workstation client hosts onto a network interface allowing the PCs to communicate with any host on the network. The router can support any combination of SLIP or PPP lines and lines dedicated to normal asynchronous devices such as terminals and modems. Refer to RFC 1055 for more information about SLIP, and RFCs 1331 and 1332 for more information about PPP.

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

Note Most asynchronous serial links have very low bandwidth. Take care to configure your system so the links will not be overloaded. Consider using default routes and filtering routing updates to prevent them from being sent on these lines.

Figure 4-1 illustrates a typical asynchronous SLIP or PPP telecommuting configuration.

Figure 4-1 Sample SLIP or PPP Telecommuting Configuration



Responding to BOOTP Requests

There is an asynchronous BOOTP server in your router. This means that SLIP and PPP clients can send BOOTP requests to the router, and the router will respond with information about the network. For example, the client can send a BOOTP request to find out what its IP address is and where the boot file is located, and the router can respond with the information.

BOOTP allows a client machine to discover its own IP address, the address of the router, and the name of a file to be loaded into memory and executed. There are typically two phases to using BOOTP: first, the client's address is determined and the boot file is selected; then the file is transferred, typically using TFTP.

BOOTP compares to RARP as follows: Reverse Address Resolution Protocol (RARP) is an older protocol that allows a client to determine its IP address if it knows its hardware address. (Refer to the chapters "Configuring IP" and "Configuring IP Routing Protocols," in the *Router Products Cnfiguration Guide*, for more information about RARP.) However, RARP is a hardware link protocol, so it can only be implemented on hosts that have special kernel or driver modifications that allow access to these raw packets. BOOTP does not require kernel modifications.

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

Asynchronous Network Connections and Routing

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

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

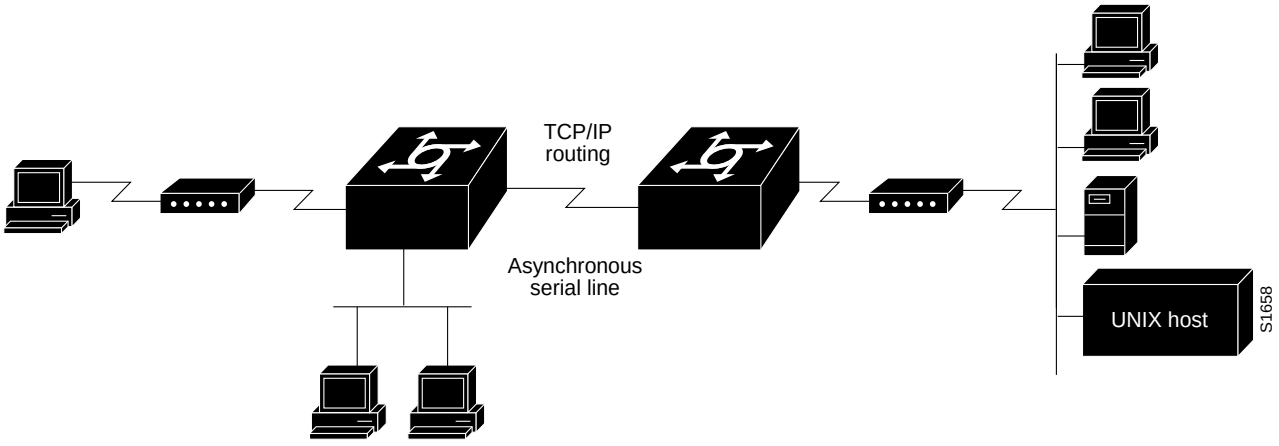
Beginning with IOS Release 10.3, your router supports protocol translation for SLIP and PPP between other network devices running Telnet, LAT, and X.25. The IOS software also supports translation for byte-oriented traffic to and from a packet assembler/disassembler (PAD). For example, you can send IP packets across a public X.25 PAD network using SLIP or PPP encapsulation when protocol translation is enabled for SLIP or PPP. For more information, refer to the chapter "Configuring Protocol Translation Sessions" in this publication.

If asynchronous dynamic routing is enabled, you can enable routing at the user level by using the **routing** keyword with the **slip** or **ppp** EXEC command.

Asynchronous interfaces offer both dedicated and dynamic address assignment, configurable hold queues and IP packet sizes, extended BOOTP requests, and permit and deny conditions for controlling access to lines. Figure 4-2 shows a sample asynchronous routing configuration.

Note Disable software flow control on SLIP and PPP lines.

Figure 4-2 Sample Asynchronous Routing Configuration



Asynchronous Interfaces and Broadcasts

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

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

Telecommuting Configuration Task List

To configure your router to support telecommuting, you must perform the first task in the following list on your asynchronous interfaces. Perform the remaining tasks to customize the asynchronous interface for your particular network environment and to monitor asynchronous connections:

- Configure Asynchronous Interfaces
- Configure Automatic Protocol Startup
- Configure Performance Parameters
- Optimize Available Bandwidth
- Specify the MTU Size of IP Packets
- Modify the IP Output Queue Size
- Specify IP Access Lists
- Configure Support for Extended BOOTP Requests
- Monitor and Maintain Asynchronous Interfaces

The steps to perform these tasks are described in the following sections. See the "Asynchronous Interface Configuration Examples" section at the end of this chapter for examples of asynchronous configuration files. Tasks are performed in global configuration mode unless otherwise specified.

Configure Asynchronous Interfaces

To configure your router to support telecommuting, configure basic functionality on your asynchronous interfaces, and then customize the interfaces for your environment. Basic configuration tasks include the following:

- Specify an Asynchronous Interface
- Configure SLIP or PPP Encapsulation
- Enable SLIP and PPP on Virtual Asynchronous Interfaces
- Specify Dedicated or Interactive Mode
- Configure the Interface Addressing Method for Remote Device
- Assign IP Addresses for Local Devices
- Enable Asynchronous Routing
- Make SLIP and PPP connections at the EXEC level if you have configured interactive mode. Refer to the *Cisco Access Connection Guide* for more information about making SLIP and PPP connections.

Specify an Asynchronous Interface

To specify an asynchronous interface, perform the following task in global configuration mode.

Task	Command
Specify an asynchronous interface.	**interface async** *number*

Configure SLIP or PPP Encapsulation

SLIP and PPP are methods of encapsulating datagrams and other network-layer protocol information over point-to-point links. To configure the default encapsulation on an asynchronous interface, perform the following task in interface configuration mode.

Task	Command
Configure PPP or SLIP encapsulation on an asynchronous line.	**encapsulation** {**slip** | **ppp**}

In order to use SLIP or PPP, the router must be configured with an IP routing protocol, with the **ip host-routing** command, or with the **vty-async** command (virtual terminal lines (VTYs) only). This configuration is done automatically if you are using old-style **slip address** commands, but you must configure it manually if you configure SLIP or PPP via the **interface async** command.

When an asynchronous interface is encapsulated with SLIP or PPP, IP fast switching is enabled. For more information on IP fast switching, refer to the "Enable Fast Switching" section later in this chapter.

Enable SLIP and PPP on Virtual Asynchronous Interfaces

The IOS software permits you to configure asynchronous protocol features, such as SLIP and PPP, on virtual terminal lines. SLIP and PPP normally function only on asynchronous interfaces, and not on virtual terminal lines. When you configure a virtual terminal line to support asynchronous protocol features, you are creating *virtual asynchronous interfaces* on the virtual terminal lines. One practical benefit of virtual asynchronous interfaces is the ability to tunnel SLIP and PPP inside of X.25, TCP, or LAT on virtual terminal lines. You tunnel SLIP and PPP using the protocol translation facility. For more information, refer to the chapter "Configuring Protocol Translation Sessions" later in this publication.

Perform the tasks in the following sections to configure and use virtual asynchronous interfaces. The first task is required; the remaining tasks are optional.

- Create Virtual Asynchronous Interfaces
- Enable Protocol Translation of SLIP and PPP on Virtual Asynchronous Interfaces
- Enable Dynamic Routing on Virtual Asynchronous Interfaces
- Enable Header Compression on Virtual Asynchronous Interfaces
- Enable Keepalive Updates on Virtual Asynchronous Interfaces
- Set an MTU on Virtual Asynchronous Interfaces
- Enable PPP Authentication on Virtual Asynchronous Interfaces
- Enable TACACS on Virtual Asynchronous Interfaces

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

Create Virtual Asynchronous Interfaces

To create a virtual asynchronous interface, perform the following task in global configuration mode:

Task	Command
Configure all virtual terminal lines to support asynchronous protocol features.	**vty-async**

Enable Protocol Translation of SLIP and PPP on Virtual Asynchronous Interfaces

One practical benefit of enabling virtual asynchronous interfaces is the ability to tunnel SLIP and PPP over X.25 PAD, thus extending remote node capability into the X.25 area. You can also tunnel SLIP and PPP over Telnet or LAT on virtual terminal lines. When tunneling SLIP and PPP over X.25, LAT, or Telnet, you do so using the protocol translation feature in the IOS software. Refer to the chapter "Configuring Protocol Translation Sessions" later in this publication for more information about protocol translation.

To translate from X.25, LAT, or Telnet to SLIP or PPP, you can use one-step protocol translation or two-step protocol translation, as follows:

- If you are translating SLIP or PPP using the one-step method, you do not need to enter the **vty-async** command. Using the **translate** command with the SLIP or PPP keywords for one-step connections automatically enables asynchronous protocol functions on virtual terminal lines. For more information about protocol translation, refer to the "Configuring Protocol Translation Sessions" chapter in this publication. For more information about using the **translate** command with the SLIP or PPP keywords, refer to the "Protocol Translation Session Commands" chapter in this publication.
- If you are translating SLIP or PPP using the two-step method, you need to first enter the **vty-async** command on a global basis on the router. Next, you perform two-step translation. For more information about two-step protocol translation, refer to the protocol translation chapter in the *Cisco Access Connection Guide*.

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

For an example of tunneling SLIP over X.25 PAD, refer to the "Configuring Protocol Translation Sessions" chapter.

Enable Dynamic Routing on Virtual Asynchronous Interfaces

To configure a virtual terminal line to support asynchronous protocol functions and route IP packets using the IGRP, RIP, and OSPF routing protocols on all virtual terminal lines, perform the following task in global configuration mode:

Task	Command
Enable dynamic routing of IP packets on all virtual terminal lines.	**vty-async dynamic-routing**

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

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

Enable Header Compression on Virtual Asynchronous Interfaces

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

You can optionally specify outgoing packets to be compressed only if TCP incoming packets on the same virtual terminal are compressed. If you do not specify this option, the router will compress all traffic. The default is no compression.

To configure a virtual terminal line to support asynchronous protocol functions and compress the headers of TCP packets, perform the following task in global configuration mode:

Task	Command
Enable header compression on IP packets on all virtual terminal lines.	**vty-async header-compression** [**passive**]

Enable Keepalive Updates on Virtual Asynchronous Interfaces

Keepalives are enabled on all virtual asynchronous interfaces by default. To change the keepalive timer or disable it on virtual asynchronous interfaces, perform the following task in global configuration mode:

Task	Command
Specify the frequency with which the IOS software sends keepalive messages to the other end of an asynchronous serial link.	**vty-async keepalive** [*seconds*]

The default interval is 10 seconds. It is adjustable in one-second increments from 0 to 32767 seconds. To turn off keepalive updates, set the value to 0. A connection is declared down after three update intervals have passed without receiving a keepalive packet.

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

Set an MTU on Virtual Asynchronous Interfaces

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

- The SLIP or PPP application at the other end only supports packets up to a certain size.
- You want to assure a shorter delay by using smaller packets.

- The host Telnet echoing takes longer than 0.2 seconds.

  For example, at 9600 baud a 1500 byte packet takes about 1.5 seconds to transmit. This delay would indicate that you want an MTU size of about 200 (1.5 seconds / 0.2 seconds = 7.5 and 1500 byte packet/ 7.5 = 200 byte packet).

To specify maximum IP packet size, perform the following task in interface configuration mode:

Task	Command
Specify the size of the largest IP packet that the virtual terminal -asynchronous line can support.	**vty-line mtu** *bytes*

TCP running on the device to which the router is connected can negotiate for a different MTU size than is configured on the router. The router performs IP fragmentation of packets larger than the specified MTU. Do not change the MTU size unless the SLIP or PPP implementation running on the host at the other end of the asynchronous line supports reassembly of IP fragments. Because each fragment occupies a spot in the output queue, it might also be necessary to increase the size of the SLIP or PPP hold queue, if your MTU size is such that you might have a high amount of fragments of packets in the output queue.

Enable PPP Authentication on Virtual Asynchronous Interfaces

You can enable Challenge Handshake Authentication Protocol (CHAP) or Password Authentication Protocol (PAP) for authentication of PPP on virtual asynchronous interfaces.

Enable CHAP

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

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

The challenge consists of 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 transmitted to the remote device.

The required response consists of two parts:

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

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

By transmitting this response, the secret is never transmitted, thus preventing other devices from stealing it and gaining illegal access to the system. Without the proper response, the remote device cannot connect to the local 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, perform the following task in global configuration mode:

Task	Command
Enable CHAP on all virtual asynchronous interfaces.	**vty-async ppp authentication chap**

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

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

Enable PAP

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

To use PAP, perform the following task in interface configuration mode:

Task	Command
Enable PAP on all virtual asynchronous interfaces.	**vty-async ppp authentication pap**

Enable TACACS on Virtual Asynchronous Interfaces

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

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

Task	Command
Enable TACACS on all virtual asynchronous interfaces.	**vty-async ppp use-tacacs**

Specify Dedicated or Interactive Mode

You can configure one or more asynchronous interfaces on your router to be in dedicated network interface mode. In dedicated mode, an interface is automatically configured for SLIP or PPP connections. There is no user prompt or EXEC level, and no end-user commands are required to initiate telecommuting connections. If you want a line to be used only for SLIP or PPP connections, configure the line for dedicated mode.

In interactive mode, a line can be used to make any type of connection, depending on the EXEC command entered by the user. For example, depending on its configuration, the line could be used for Telnet or XRemote connections, or SLIP or PPP encapsulation. The user is prompted for an EXEC command before a connection is initiated.

This section describes the following tasks:

- Configure Dedicated Network Mode
- Return a Line to Interactive Mode

Configure Dedicated Network Mode

You can configure an asynchronous interface to be in dedicated network mode. When the interface is configured for dedicated mode, you cannot change the encapsulation method, address, or other parameters.

To configure an interface for dedicated network mode, perform the following task in interface configuration mode.

Task	Command
Place the line into dedicated asynchronous network mode.	**async mode dedicated**

Refer to the chapter "Managing the System" in the *Router Products Configuration Guide* for more information about automatic dialing using DTR.

Return a Line to Interactive Mode

After a line has been placed in dedicated mode, perform the following task in interface configuration mode to return it to interactive mode.

Task	Command
Return the line to interactive mode.	**async mode interactive**

By default, no asynchronous mode is configured. In this state, the line is not available for inbound networking because the SLIP and PPP connections are disabled.

Configure the Interface Addressing Method for Remote Device

You can control whether addressing is dynamic (the user specifies the address at the EXEC level when making the connection), or whether default addressing is used (the address is forced by the system). If you specify dynamic addressing, the router must be in interactive mode and you will enter the address at the EXEC level.

It is common to configure an asynchronous interface to have a default address and to allow dynamic addressing. With this configuration, the choice between the default address or a dynamic addressing is made by the user when they enter the **slip** or **ppp** EXEC command. If you enter an address, it is used, and if you enter the **default** keyword, the default address is used.

This section describes the following tasks:

- Assign a Default Asynchronous Address
- Allow an Asynchronous Address to Be Assigned Dynamically

Assign a Default Asynchronous Address

Perform the following task in interface configuration mode to assign a permanent default asynchronous address:

Task	Command
Assign a default IP address to an asynchronous interface.	**async default ip address** *address*

Use the **no** form of this command to disable the default address. If the server has been configured to authenticate asynchronous connections, you are prompted for a password after entering the SLIP or PPP EXEC command before the line is placed into asynchronous mode.

The assigned default address is implemented when the user enters the **slip default** or **ppp default** EXEC command. The transaction is validated by the TACACS server (when enabled) and the line is put into network mode using the address that is in the configuration file.

Configuring a default address is useful when the user is not required to know the IP address to gain access to a system; for example, users of a server that is available to many students on a campus. Instead of requiring each user to know an IP address, they need only enter the **slip default** or **ppp default** EXEC command and let the server select the address to use. See the *Cisco Access Connection Guide* for more information about the **slip** and **ppp** EXEC commands.

Allow an Asynchronous Address to Be Assigned Dynamically

When a line is configured for dynamic assignment of asynchronous addresses, you enter the **slip** or **ppp** EXEC command and are prompted for an address or logical host name. The address is validated by the Terminal Access Controller Access System (TACACS), when enabled, and the line is assigned the given address and put into asynchronous mode. Assigning asynchronous addresses dynamically is also useful when you want to assign set addresses to users. For example, an application on a personal computer that automatically dials in using SLIP and polls for electronic mail messages can be set up to dial in periodically and enter the required IP address and password.

To assign asynchronous addresses dynamically, perform the following task in interface configuration mode:

Task	Command
Allow the IP address to be assigned when the protocol is initiated.	**async dynamic address**

The dynamic addressing features of the internetwork allow packets to get to their destination and back regardless of the router or network they are sent from. For example, if a host such as a laptop computer moves from place to place it can keep the same address no matter where it is dialing in from.

Logical host names are first converted to uppercase and then sent to the TACACS server for authentication.

Assign IP Addresses for Local Devices

The local address is set using the **ip address** or **ip unnumbered** command.

IP addresses identify locations to which IP datagrams can be sent. You must assign each router interface an IP address. See the *Internetworking Technology Overview* publication for detailed information on IP addresses.

To assign an IP address to a network interface on the router, perform the following task in interface configuration mode:

Task	Command
Set an IP address for an interface.	**ip address** *address mask* [**secondary**]

A subnet mask identifies the subnet field of a network address. Subnets are described in the *Internetworking Technology Overview* publication.

Conserve Network Addresses

When asynchronous routing is enabled, you might find it necessary to conserve network addresses by configuring the asynchronous interfaces as *unnumbered*. An unnumbered interface does not have an address. Network resources are therefore conserved because fewer network numbers are used and routing tables are smaller.

To configure an unnumbered interface, perform the following task in interface configuration mode.

Task	Command
Configure the asynchronous interface to be unnumbered.	**ip unnumbered** *type number*

Whenever the unnumbered interface generates a packet (for example, a routing update), it uses the address of the specified interface as the source address of the IP packet. It also uses the address of the specified interface to determine which routing processes are sending updates over the unnumbered interface.

You can use the IP unnumbered feature on the router whether or not the system on the other end of the asynchronous link supports this feature. The IP unnumbered feature is transparent to the other end of the link because each system bases its routing activities on information in the routing updates it receives and on its own interface address on the link.

Enable Asynchronous Routing

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

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

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

Configure Automatic Protocol Startup

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

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

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

The **autoselect** command permits the router to allow an appropriate process to start automatically when a starting character is received. The router detects either a Return character, which is the start character for an EXEC session, or the start character for the ARA protocol.

Configure Performance Parameters

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

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

Compress TCP Packet Headers

You can compress the headers of your TCP/IP packets in order to reduce their size, thereby increasing performance. Header compression is particularly useful on networks with a large percentage of small packets, such as those supporting many Telnet connections. This feature only compresses the TCP header, so it has no effect on UDP packets or other protocol headers. The TCP header compression technique, described fully in RFC 1144, is supported on serial lines using HDLC or PPP encapsulation. You must enable compression on both ends of a serial connection.

You can optionally specify outgoing packets to be compressed only if TCP incoming packets on the same interface are compressed. If you do not specify this option, the router will compress all traffic. The default is no compression.

You can also specify the total number of header compression connections that can exist on an interface. You should configure one connection for each TCP connection through the specified interface.

To enable compression, perform either of the following optional tasks in interface configuration mode:

Task	Command
Enable TCP header compression.	**ip tcp header-compression** [**passive**]
Specify the total number of header compression connections that can exist on an interface.	**ip tcp compression-connections** *number*[1]

1. This command is documented in the "IP Commands" chapter of the *Router Products Command Reference* publication.

Note When compression is enabled, fast switching is disabled. Fast processors can handle several fast interfaces, such as T1s, that are running header compression. However, you should think carefully about your network's traffic characteristics before compressing TCP headers. You might want to use the monitoring commands to help compare network utilization before and after enabling header compression.

Set the TCP Connection Attempt Time

You can set the amount of time the router will wait to attempt to establish a TCP connection. In previous versions of the IOS software, the system would wait a fixed 30 seconds when attempting to do so. This amount of time is not sufficient in networks that have dial-up asynchronous connections, such as a network consisting of dial-on-demand links that are implemented over modems because it will affect your ability to Telnet over the link (from the router) if the link must be brought up.

Because the connection attempt time is a host parameter, it does not pertain to traffic going through the router, just to traffic originated at the router.

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

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

1. This command is documented in the "IP Commands" chapter of the *Router Products Command Reference* publication.

Enable Fast Switching

Fast switching involves the use of a high-speed switching cache for IP routing. With fast switching, destination IP addresses are stored in the high-speed cache so that some time-consuming table lookups need not be done. Our routers generally offer better packet transfer performance when fast switching is enabled.

To enable or disable fast switching, perform the following tasks in interface configuration mode:

Task	Command
Enable fast-switching (use of a high-speed route cache for IP routing).	**ip route-cache**[1]
Disable fast switching and enable load balancing on a per-packet basis.	**no ip route-cache**[1]

1. These commands are documented in the "IP Commands" chapter of the *Router Products Command Reference* publication.

Control Route Cache Invalidation

The high-speed route cache used by IP fast switching is invalidated when the IP routing table changes. By default, the invalidation of the cache is delayed slightly to avoid excessive CPU load while the routing table is changing.

To control route cache invalidation, perform the following tasks in global configuration mode as needed for your network:

Task	Command
Allow immediate invalidation of the cache.	**no ip cache-invalidate-delay**[1]
Delay invalidation of the cache.	**ip cache-invalidate-delay** [*minimum maximum quiet threshold*][1]

1. These commands are documented in the "IP Commands" chapter of the *Router Products Command Reference* publication.

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

Optimize Available Bandwidth

Asynchronous lines have relatively low bandwidth and can easily be overloaded, resulting in slow traffic across these lines.

To optimize available bandwidth, perform any of the following tasks:

- Configure Header Compression
- Force Header Compression at the EXEC Level

Configure Header Compression

One way to optimize available bandwidth is by using TCP header compression. Van Jacobson TCP header compression (defined by RFC 1144) can increase bandwidth availability between two and five times when compared to lines not using header compression. Theoretically, it can improve bandwidth availability by a ratio of seven to one.

To configure header compression, perform the following task in interface configuration mode:

Task	Command
Configure Van Jacobson TCP header compression on the asynchronous link.	**ip tcp header-compression** [**on** | **off** | **passive**]

Force Header Compression at the EXEC Level

On SLIP interfaces, you can force header compression at the EXEC prompt on a line on which header compression has been set to passive. This allows more efficient use of the available bandwidth and does not require entering privileged configuration mode.

To implement header compression, perform the following task in interface configuration mode:

Task	Command
Allow status of header compression to be assigned at the user level.	**ip tcp header-compression passive**

For PPP interfaces, the **passive** option functions the same as the **on** option.

See the *Cisco Access Connection Guide* for information about the **slip** and **ppp** EXEC commands. You cannot force header compression if header compression on the asynchronous interface is off.

Specify the MTU Size of IP Packets

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

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

  For example, at 9600 baud a 1500 byte packet takes about 1.5 seconds to transmit. This delay would indicate that you want an MTU size of about 200 (1.5 seconds / 0.2 seconds = 7.5 and 1500 byte packet/ 7.5 = 200 byte packet).

To specify maximum IP packet size, perform the following task in interface configuration mode:

Task	Command
Specify the size of the largest IP packet that the asynchronous line can support.	**ip mtu** *bytes*

The MTU size can be negotiated by TCP, regardless of the asynchronous interface settings. In other words, TCP running on the device to which the router is connected can negotiate for a different MTU size than is configured on the router. The router performs IP fragmentation of packets larger than the specified MTU. Do not change the MTU size unless the SLIP or PPP implementation running on the host at the other end of the asynchronous line supports reassembly of IP fragments. Because each fragment occupies a spot in the output queue, it might also be necessary to increase the size of the SLIP or PPP hold queue, if your MTU size is such that you might have a high amount of fragments of packets in the output queue.

Modify the IP Output Queue Size

The IP output queue stores packets received from the network that are waiting to be sent to the asynchronous client. You can limit the size of the IP output queue to enhance performance by performing the following task in interface configuration mode:

Task	Command
Change the size of the IP output hold queue.	**hold-queue** *packets*

Specify IP Access Lists

Access lists allow the system administrator to control the hosts that a PC can access through a router. Separate access lists can be defined for asynchronous and for other connections.

The tasks described in this section are as follows:

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

Refer to the chapter "Configuring IP," in the *Router Products Configuration Guide* publication, for information about defining IP access lists.

To define an access list for packets from the IP host, perform the following task in interface configuration mode:

Task	Command
Configure an access list for packets from the IP host.	**ip access-group** *access-list-number* **in**

To define an access list for packets to the IP host, perform the following task in interface configuration mode:

Task	Command
Configure an access list for packets being sent to the IP host.	**ip access-group** *access-list-number* **out**

Configure Support for Extended BOOTP Requests

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

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

Monitor and Maintain Asynchronous Interfaces

This section describes the following monitoring and maintenance tasks:

- Monitor and maintain asynchronous activity
- Debug asynchronous interfaces
- Debug PPP

To monitor and maintain asynchronous activity, perform one or more of the following tasks in privileged EXEC mode:

Task	Command
Return a line to its idle state.	**clear line** *line-number*
Display parameters that have been set for extended BOOTP requests.	**show async bootp**
Display statistics for asynchronous activity.	**show async status**
Display the status of asynchronous line connections.	**show line** [*line-number*]

To debug asynchronous interfaces, perform the following task in privileged EXEC mode:

Task	Command
Displays errors, changes in interface state, and log input and output.	**debug async** {**framing** | **state** | **packets**}

To debug PPP links, perform the following tasks in privileged EXEC mode:

Task	Command
Enable debugging of PPP protocol negotiation process.	**debug ppp negotiation**
Display PPP protocol errors.	**debug ppp error**
Display PPP packets sent and received.	**debug ppp packet**
Display errors encountered during remote or local system authentication.[1]	**debug ppp chap**

1. Refer to the chapter “Configuring DDR” in the *Router Products Configuration Guide* publication for more information about the Challenge Handshake Authentication Protocol (CHAP).

Asynchronous Interface Configuration Examples

This section contains asynchronous configuration examples. Each configuration is designed to illustrate different communication requirements.

- Dedicated Asynchronous Interface Configuration Example
- Restricted Access on an Asynchronous Interface Example
- Asynchronous Routing and Dynamic Addressing Configuration Example
- TCP Header Compression Configuration Example
- Conserving Network Addresses Using the IP Unnumbered Feature Example
- Configuring Routing on a Dedicated Dial-In Router Example
- Configuring an Asynchronous Interface as the Only Network Interface Example
- Configuring IGRP Example
- Configuring an Interface Example

Dedicated Asynchronous Interface Configuration Example

The following example assigns an IP address to an asynchronous interface and places the line in dedicated network mode. Setting the stop bit to 1 is a performance enhancement.

```
line 20
location Department PC Lab
stopbits 1
speed 19200
!
interface async 20
async default ip address 182.32.7.51
async mode dedicated
```

Note The interface number is the same as the absolute line number, in decimal format.

Restricted Access on an Asynchronous Interface Example

The following example assumes that users are restricted to certain servers designated as asynchronous servers, but that normal terminal users can access anything on the local network.

```
! access list for normal connections
access-list 1 permit 131.108.0.0 0.0.255.255
!
access-list 2 permit 131.108.42.55
access-list 2 permit 131.108.111.1
access-list 2 permit 131.108.55.99
!
interface async 6
async dynamic address
ip access-group 1 out
ip access-group 2 in
```

Asynchronous Routing and Dynamic Addressing Configuration Example

The following example shows a simple configuration that allows routing and dynamic addressing. With this configuration, if the user specifies **/routing** in the EXEC **slip** or **ppp** command, routing protocols will be sent and received.

```
interface async 6
async dynamic routing
async dynamic address
```

TCP Header Compression Configuration Example

The following example configures async interface 7 with a default IP address, allowing header compression if it is specified in the **slip** or **ppp** connection command entered by the user or if the connecting system sends compressed packets.

```
interface async 7
ip address 150.136.79.1
async default ip address 150.136.79.2
ip tcp header-compression passive
```

Conserving Network Addresses Using the IP Unnumbered Feature Example

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

```
interface ethernet 0
ip address 128.66.1.1 255.255.255.0
!
interface async 6
ip unnumbered ethernet 0
async dynamic routing
! default address is on the local subnet
async dynamic address
async default ip address 128.66.1.2
ip tcp header-compression passive
```

The following example shows how the IP unnumbered configuration works. Although the user assigned an address, the system response shows the interface as unnumbered, and the address typed by the user will be used only in response to BOOTP requests.

```
router> slip /compressed 1.1.1.1
Password:
Entering async mode.
Interface IP address is unnumbered, MTU is 1500 bytes.
Header compression is On.
```

Configuring Routing on a Dedicated Dial-In Router Example

In the following example, the router is set up as a dedicated dial-in router. Interfaces are configured as IP unnumbered to conserve network resources, primarily IP addresses.

```
ip routing
interface ether 0
ip address 1.0.1.1 255.255.255.0
!
interface async 1
ip unnumbered ethernet 0
async dynamic routing
! The addresses assigned with SLIP or PPP EXEC commands are not used except
```

```
! to reply to BOOTP requests.
! Normally, the routers dialing in will have their own address
! and not use BOOTP at all.
async default ip address 1.0.2.1
!
interface async 2
ip unnumbered ethernet 0
async default ip address 1.0.5.1
ip tcp header-compression passive
async mode dedicated
!
! run RIP on the asynchronous lines, because few implementations of SLIP
! understand IGRP. Run IGRP on the ethernet (and in the local network).
!
router igrp 109
network 1.0.0.0
! send routes from the asynchronous lines on the production network.
redistribute RIP
! don't send IGRP updates on the async interfaces
passive-interface async 1
!
router RIP
network 1.0.0.0
redistribute igrp
passive-interface ethernet 0
! consider filtering everything except a default route from the routing
! updates sent on the (slow) asynchronous lines
distribute-list 1 out
ip unnumbered async 2
async dynamic routing
```

Configuring an Asynchronous Interface as the Only Network Interface Example

In the following example, one of the asynchronous lines is used as the only network interface. The router is used primarily as a router, but is at a remote location and dials into the central site for its only network connection.

```
ip default-gateway 1.0.0.2
interface ethernet 0
shutdown
interface async 1
async dynamic routing
ip tcp header-compression on
async default ip address 1.0.5.1
async mode dedicated
ip address 1.0.0.1 255.255.255.0
```

Configuring IGRP Example

In the following example, only the IGRP TCP/IP routing protocol is running; it is assumed that the systems that are dialing in to use routing will either support IGRP or have some other method (for example, a static default route) of determining that the router is the best place to send most of its packets.

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

```
ip tcp header-compression passive
ip unnumbered ethernet 0

line 1
modem ri-is-cd
```

Configuring an Interface Example

The following configuration shows interface and line configuration. The interface is configured with access lists, passive header compression and a default address. The line is configured for TACACS authentication.

```
interface async 1
ip access-group 1 in
ip access-group 1 out
ip tcp header-compression passive
async default ip address 128.148.176.201

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

CHAPTER 5

Configuring XRemote

This chapter describes the X Window System and how to configure your router to support XRemote connections.

For a complete description of the commands mentioned in this chapter, refer to the chapter "XRemote Configuration Commands" later in this publication. For information about establishing XRemote connections, refer to the *Cisco Access Connection Guide*.

Cisco's Implementation of the X Window System

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

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

X and the Client–Server Model

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

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

How XRemote Works

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

There are two basic parts to XRemote:

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

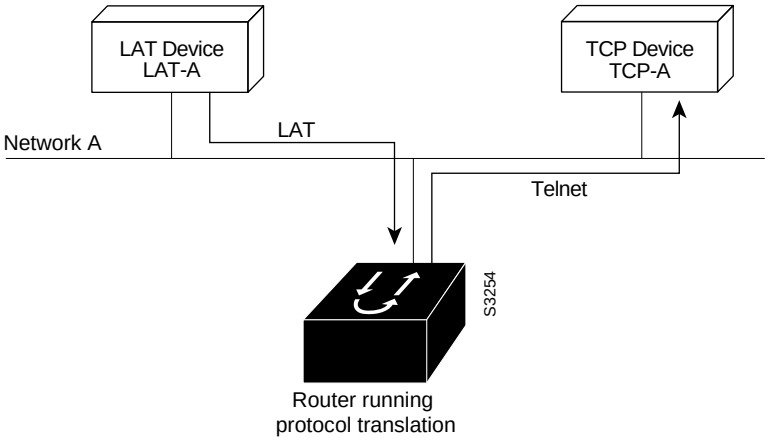
These two helper processes communicate with each other using the optimized XRemote protocol. The client-side helper communicates with X clients using the standard X protocol. The server-side helper communicates with the server using standard X. The server-side helper might operate as part of the X server or it might be external and accessed across the network; for example, the server-side helper can operate in a router at your house or work site. If the server-side helper is in the X terminal, it must have XRemote PROMs installed.

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

Connection Capability

Cisco's implementation of XRemote is fully compatible with the NCD XRemote protocol. Figure 5-1 illustrates an XRemote connection between an X terminal and a router. In Figure 5-1, the server-side helper is running on the X terminal and the client-side helper is running on the router.

Figure 5-1 XRemote Session from an X Display Server Running XRemote



Remote Access to Fonts

Remote access to fonts is provided by using one of the following protocols:

- Industry-standard protocol for transporting X traffic over Transmission Control Protocol/Internet Protocol (TCP/IP) networks
- Digital Equipment Corporation protocol for transporting X traffic over Local Area Transport (LAT) networks
- Internet standard Trivial File Transfer Protocol (TFTP) for TCP/IP networks
- Digital Equipment Corporation protocol for font access via LAT

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

XRemote Configuration Task List

To make a host connection using NCD's XRemote feature and the router, complete the basic configuration steps in the following sections, as needed:

- Attach a Modem to the Auxiliary Port
- Set Up X Terminal Parameters
- Define a Font Server
- Select the Fonts
- Increase the Internal Buffer Size
- Set the Number of Font Loader Retries
- Monitor XRemote Activity

See the end of this chapter for an XRemote configuration file example.

Attach a Modem to the Auxiliary Port

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

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

Refer to the *Router Products Configuration Guide* for more information about configuring modems.

Set Up X Terminal Parameters

The X server for the X terminal and the Network and Serial parameters for the X terminal must be configured as described in the publications for the specific X terminal you are using. In general, the X terminal configuration determines the mode of operation for the terminal, the source of font information, and the source of remote configuration information (when applicable).

Define a Font Server

If you intend to download fonts to the X terminal, you must set the font loader host identification. To do this, perform the following global configuration task:

Task	Command
Specify a TFTP font server as the source for fonts.	**xremote tftp host** [*host-name*]

Select the Fonts

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

Perform the tasks in the following sections to select fonts:

- Access Nonresident Fonts via TFTP
- Select DECwindows Fonts

Access Nonresident Fonts via TFTP

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

An X server can display only the fonts it finds in the directories in its font path. The X server's default font path includes only the built-in fonts. To access fonts stored on a host, you must add the host's font directories to the X server's font path. To do this, use the UNIX command **xset** with the **fp+** argument to add fonts to the end of the server's font path.

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

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

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

Select DECwindows Fonts

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

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

Execute **xset**, or the application to issue an XSetFontPath request, to set the following path:

/LAT/*SERVICE*

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

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

Increase the Internal Buffer Size

When the X terminal requests that a font file be loaded, the router must first load the font file into an internal buffer before passing it to the X terminal. The default value for this buffer is 70,000 bytes, which is adequate for most font files, but the size can be increased as necessary for nonstandard font files.

To change the buffer size, perform the following global configuration task:

Task	Command
Set the buffer size used for loading font files.	**xremote tftp buffersize** [*buffer-size*]

This task can be performed for both TFTP and LAT font access.

Set the Number of Font Loader Retries

You might need to increase the number of times that the font loader will try to load the fonts. This is particularly important when the font servers are known to be heavily loaded.

To set the TFTP font loader retries, perform the following global configuration task:

Task	Command
Set the number of retries by the TFTP font loader.	**xremote tftp retries** [*retries*]

Monitor XRemote Activity

To check the status of XRemote connections, perform one or more of the following tasks at the EXEC prompt:

Task	Command
Display current connections and monitor traffic.	**show xremote**
Display traffic and line statistics.	**show xremote line** *number*

XRemote Configuration File Example

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

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

CHAPTER 6

Configuring Protocol Translation Sessions

This chapter describes how to configure protocol translation sessions. It assumes you understand how to use the IOS software. It provides procedures for specifying system-wide facilities, as well as application examples. Before continuing with this chapter, be sure that you are familiar with the information provided in the Local Area Transport (LAT), TN3270, Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP), and XRemote configuration chapters in this publication. For information about X.25 and Transmission Control Protocol/Internet Protocol (TCP/IP, which includes TCP, SLIP and PPP, and Telnet), refer to the *Router Products Configuration Guide* and the *Router Products Command Reference* publications.

For a complete description of the commands in this chapter, see the "Protocol Translation Session Commands" chapter later in this publication. For information about making connections and establishing translation sessions, see the *Cisco Access Connection Guide*.

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.

Cisco's Implementation of Protocol Translation

The IOS software attempts to provide transparent protocol translation between systems running disparate protocols. The software fully supports two-way virtual terminal protocol translation between nodes running X.25, LAT, SLIP or Point-to-Point Protocol (PPP), and Telnet.

The IOS software supports limited translation for XRemote (to X.25 PAD environments) and TN3270 (to LAT, X.25, and Telnet environments). However, there is no translation between other services such as file-transfer protocols. The protocol translation software allows 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 target host. You can use either of two methods to accomplish this:

- One-step method—This method provides full transparent protocol translation by means of the **translate** global configuration command.This command defines a connection and the translation protocols in the configuration file. Then you can make single-step connections to remote resources.
- Two-step method—Actual connections between devices are made by first establishing a connection to the router running protocol translation, then by establishing the outgoing connection to the remote destination.

A router running protocol translation uses the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) Recommendation X.25 for transferring raw data over X.25 networks. The X.25 software supports both commercial and defense data network (DDN) versions. Such a router also supports X.25 as a transport mechanism for IP packets, and X.3- and X.29-based PAD connections. This allows the routers with protocol translation to connect to an X.25 public data network (PDN). This X.25 connection allows transport of TCP/IP packets across the X.25 packet-switching network in the same way as would occur on a standalone protocol translator.

Note The ITU-T carries out the functions of the former Consultative Committee for International Telegraph and Telephone (CCITT).

Routers without protocol translation software do not support an X.25 PAD function, so they cannot communicate with hosts directly connected to the X.25 PDN. Routers with protocol translation encapsulate TCP/IP packets in X.25 packets for transfer over a packet-switching network. These packets can be received by another router configured with protocol translation software. Only protocol translation software includes PAD capabilities, but other internetworking devices can communicate with routers running such software using the TCP/IP or LAT protocols. Protocol translation software can translate TCP/IP and LAT into X.25, and then communicate with X.25 hosts. Protocol translation software supports all PAD standards (X.28, X.29, and X.3). The connection to the packet-switching network is through a synchronous line.

Connections to a PAD are made using EXEC commands. You can configure PAD parameter profiles that can be used to set PAD parameters with other commands, and you can configure access lists to control X.25 network access. Both these features use the message fields defined in Recommendation X.29, which describes procedures for exchanging data between PADs or between a PAD and a DTE.

Protocol Translation Configuration Task List

Perform the tasks in the following sections, as appropriate, when setting up protocol translation:

- Create a Virtual Asynchronous Interface to Translate SLIP or PPP
- Configure One-Step Protocol Translation
- Configure Two-Step Protocol Translation
- Change the Number of Supported Translation Sessions

Perform the tasks in the following sections to apply X.29 PAD connections under X.25:

- Create X.29 Access Lists
- Create an X.29 Profile Script

See the section “Protocol Translation Session Configuration Examples” at the end of this chapter for protocol translation application examples.

Create a Virtual Asynchronous Interface to Translate SLIP or PPP

If you want to translate SLIP or PPP (on outgoing connections only) using the two-step protocol translation method, you first need to enable asynchronous protocol functions on all virtual terminal lines that have been created for the router.

Note You must use the two-step method for translations of TN3270 and XRemote.

SLIP and PPP function only on asynchronous interfaces, or on virtual terminal lines set up for asynchronous protocol functions. A virtual terminal line set up for asynchronous protocol functions is called a *virtual asynchronous interface*. For more information about virtual asynchronous interfaces, refer to the "Configuring SLIP and PPP" chapter in this publication.

Enabling SLIP and PPP on virtual asynchronous interfaces permits you to run SLIP and PPP in an X.25 PAD environment, thus extending remote node capability into the X.25 area. You can also run SLIP and PPP in a Telnet or LAT environment.

To translate from X.25, LAT, or Telnet to SLIP or PPP using the one-step method, you do not need to enter any special commands. Simply use the **translate** command with the SLIP or PPP keywords for one-step connections. The **translate** command automatically enables asynchronous protocol functions on one virtual terminal line at a time.

To create a virtual asynchronous interface to enable translation of SLIP and PPP, perform the following task in global configuration mode:

Task	Command
Create a virtual asynchronous interface for translating SLIP and PPP.	**vty-async**[1]

1. This command is described in the "SLIP and PPP Configuration Commands" chapter in this publication.

For more information about configuring virtual asynchronous interfaces, refer to the "Configuring SLIP and PPP" chapter.

To configure the two-step method of protocol translation, see the configuration task list in the *Cisco Access Connection Guide*.

Configure One-Step Protocol Translation

The one-step method of protocol translation invokes a *translation session* process to provide fully transparent protocol conversion. In this method, the router running protocol translation software masquerades as two or more hosts on the same network. When a connection is made to such a router, the router determines which host the connection is for and what protocol that host is using. The router then establishes a new network connection using the networking protocol required by that host. This network connection is more efficient and allows the router to act upon greater knowledge of the protocols in use because the router acts as a network connection rather than a terminal.

One disadvantage of using the one-step method is that the initiating computer or user does not know that two networking protocols are being used. This means that parameters of the foreign network protocols cannot be changed once the connections are established. The exception to this limitation is the set of parameters common to both networking protocols. Any parameter in this set can be changed from the first host to the final destination.

To 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 and PPP are now also supported.
- The connection features and global options—You can specify additional features for the connection that allows, for example, incoming call addresses to match access-list conditions or that limit the number of users that can make the connection.

To perform one-step protocol translation of LAT, X.25, Telnet, or SLIP/PPP, enter the following command in global configuration mode:

Task	Command
Configure protocol translation.	**translate** *protocol incoming-address* [*in-options*] *protocol outgoing-address* [*out-options*] [*global-options*]

SLIP and PPP can only be translated on outgoing connections.

Note To translate TN3270 or XRemote, you must use the two-step method.

For examples of using the one-step method, see the section "Protocol Translation Session Configuration Examples" later in this chapter.

Configure Two-Step Protocol Translation

In the two-step method, perform the following two steps:

Step 1 Establish the incoming connection directly to the router running protocol translation software.

Step 2 Establish the outgoing connection from the router running protocol translation software to another network host.

Specifically, once the transmission protocols—LAT, X.25, TCP/IP, SLIP, or PPP—are configured, all that is required for the two-step protocol translation method is to enter the EXEC connection commands—**lat**, **pad**, **telnet**, **slip**, **ppp**—first to the router, then to the outgoing device.

Protocol translation software supports the two-step method in both directions (for example, from Telnet to PAD, and vice versa).

In general, you use the two-step method when you want to use a router running protocol translation software as a *general-purpose gateway* between two types of networks (for example, X.25 PDN and TCP/IP). Instead of configuring every possible connection via embedded **translate** commands, the two-step method allows you greater flexibility in terms of connecting to network resources accessible via the router running protocol translation software.

The two-step method also allows additional security when TACACS and password protection is enabled. These security features are described in *Router Products Configuration Guide*, in the chapter "Managing the System." Additionally, if you use the two-step method to connect, you can change X.25 PAD and local terminal parameters dynamically during a session.

With the two-step connection method, you can individually 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 router running protocol translation

software. The difference is that you do not encounter the wiring complexity, unreliability, management problems, and performance bottlenecks that occur when two separate devices are connected via asynchronous serial lines.

Change the Number of Supported Translation Sessions

The number of protocol translation sessions supported by the IOS Release 10.3 software depends on the router model. On a Cisco 4000 or Cisco 4500, the maximum number of protocol translation sessions is 180 whether or not routing is enabled. On a Cisco 2500 or Cisco 3000, the maximum number of protocol translation sessions is 100 sessions, or 32 sessions if routing is enabled.

The number of protocol translation sessions is tied to the number of virtual terminal lines set up on the router running protocol translation software. That is, if such a router has ten virtual terminal lines, you can have up to ten protocol translation sessions. The default number of virtual terminal lines is five (lines 0 through 4). To increase the number of lines, and thus the maximum number of protocol translation sessions, perform the following task in line configuration mode:

Task	Command
Increase the number of virtual terminal lines, and thus, the maximum number of protocol translation sessions.	**line vty** *number*[1]
Decrease the number of virtual terminal lines, and thus, the maximum number of protocol translation sessions.	**no line vty** *number*

1. This command is documented in the "Terminal Lines and Modem Support Commands" chapter in the *Router Products Command Reference* publication.

Note Increasing the number of protocol translation sessions while routing or bridging 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 session, and the speed of the link. If you increase to the maximum number of sessions and have problems with memory, you might need to decrease the number of protocol translation sessions.

Create X.29 Access Lists

The protocol translation software provides access lists, which make it possible to limit access to a router running protocol translation software 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 following tasks:

Step 1 Create an access list.

Step 2 Apply an access list to a virtual line or include it in a **translate** command.

These tasks are described in the following sections.

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.

Create an Access List

To specify the access conditions, perform the following global configuration task:

Task	Command
Restrict incoming and outgoing connections between a particular virtual terminal line (into a Cisco device) and the addresses in an access list.	**x29 access-list** *access-list-number* {**permit** | **deny**} *regular-expression*

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.

Apply an Access List to a Virtual Line

To apply an access list to a virtual line, perform the following tasks in line configuration mode:

Task	Command
Restrict incoming and outgoing connections between a particular virtual terminal line (into a Cisco device) and the addresses in an access list.	**access-class** *number* **in**[1]

1. This command is documented in the "LAT Configuration Commands" chapter.

The access-list number is used both for incoming TCP access and for incoming PAD access. For TCP access, the router running protocol translation software uses the defined IP access lists. For incoming PAD connections, the same X.29 access list is used. If you want to have 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, see the section "X.29 Access List Example" at the end of this chapter.

Create an X.29 Profile Script

You can create an X.29 profile script for use by the **translate** command. When an X.25 connection is established, the router running protocol translation software then acts as if an X.29 SET PARAMETER packet had been sent that contained the parameters and values set by this command.

To create an X.29 profile script, perform the following global configuration task:

Task	Command
Create an X.29 profile script.	**x29 profile** *name parameter*:*value* [*parameter*:*value*]

Define X.25 Host Names

To define symbolic host names, perform the following task in global configuration mode:

Task	Command
Define a symbolic host name.	**x25 host** *name x.121-address* [**cud** *call-user-data*]

Protocol Translation Session Configuration Examples

The following sections present examples of configuring protocol translation sessions. The environment used in these examples is illustrated in Figure 6-1.

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

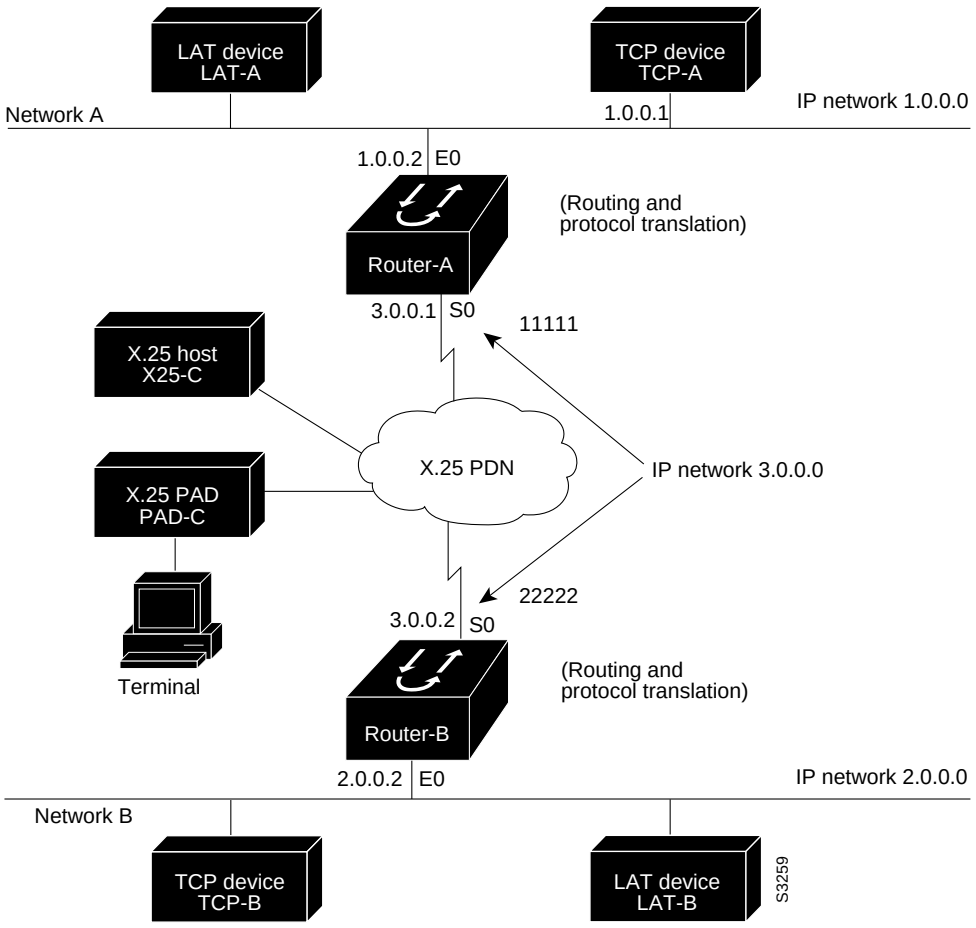
In these examples, a LAT device can be a LAT terminal server or LAT host, and a TCP device can be either a TCP terminal server or TCP host.

Note In the application illustrations that follow throughout 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 (as shown for all elements in Figure 6-1).

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 6-1. Refer to the chapter "Configuring LAT," earlier in this publication, for more information about LAT. For information on configuring X.25, refer to the *Router Products Configuration Guide*.

Figure 6-1 Summary Diagram Showing Routers with Protocol Translation Software



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 Device Router-A

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

```
interface ethernet 0
ip address 1.0.0.2 255.255.0.0
!
! 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 3.0.0.1 255.255.0.0
x25 map ip 3.0.0.2 22222 broadcast
!
! Set up IP routing
router igrp 100
network 1.0.0.0
network 3.0.0.0
!
! 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 1.0.0.2 3.0.0.1
ip host TCP-A 1.0.0.1
ip host TCP-B 2.0.0.1
ip host Router-B 3.0.0.2 2.0.0.2
```

Configuration for Device Router-B

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

```
interface ethernet 0
ip address 2.0.0.2 255.255.0.0
!
! 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 3.0.0.2 255.255.0.0
x25 map ip 3.0.0.2 11111 broadcast
!
! Set up IP routing
router igrp 100
network 2.0.0.0
network 3.0.0.0
!
! 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 3.0.0.1 1.0.0.2
ip host TCP-A 1.0.0.1
ip host TCP-B 2.0.0.1
ip host Router-B 2.0.0.2 3.0.0.2
```

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.

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
```

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:

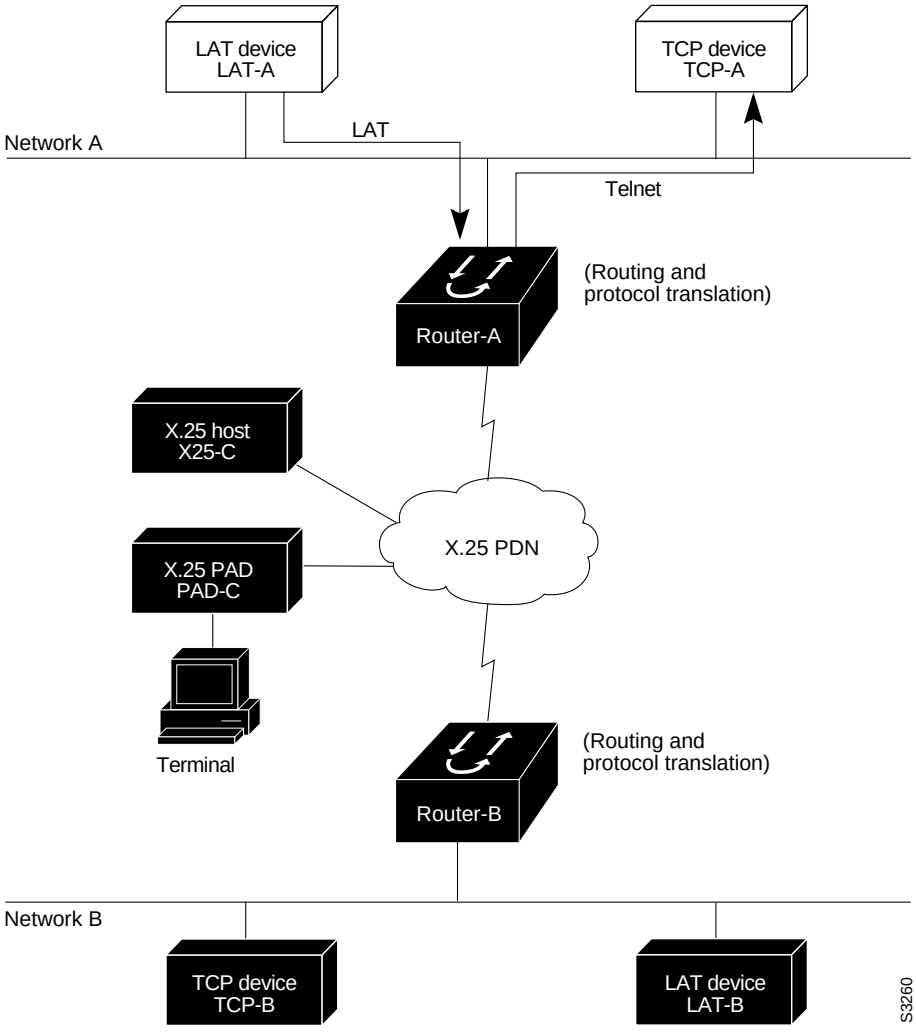
```
no line vty 10
```

Local LAT-to-TCP Example

A router running protocol translation software can translate between LAT and Telnet traffic to allow communication among resources in these protocol environments. In Figure 6-2, the LAT device on Network A (LAT-A) is shown connecting to a device running Telnet (TCP-A).

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

Figure 6-2 Local LAT-to-TCP Translation



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

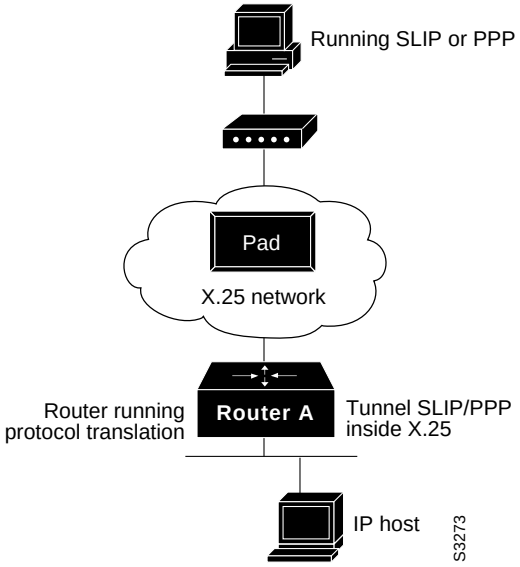
```
! Translate LAT connections to TCP for connectivity to TCP-A
translate lat TCP-A tcp TCP-A
! Optional additional commands
lat service TCP-A ident Protocol Translation to TCP-A
```

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 servers on the local network.

Tunneling SLIP or PPP Inside X.25 Example

A router running protocol translation software can translate between X.25 and PPP traffic to allow communication among resources in these protocol environments. In Figure 6-3, the PC establishes a SLIP or PPP session with the router through an X.25 network using CHAP authentication. The router then establishes a connection to IP host and passes traffic from the PC running SLIP or PPP to IP host.

Figure 6-3 Tunneling SLIP or PPP inside X.25



The following configuration tunnels PPP inside X.25 packets from the PPP client to the virtual asynchronous interface with IP address 1.0.0.4. Routing and CHAP authentication are enabled for the PPP session. With the router connected to IP host, the PC running SLIP or PPP can now communicate with the IP host.

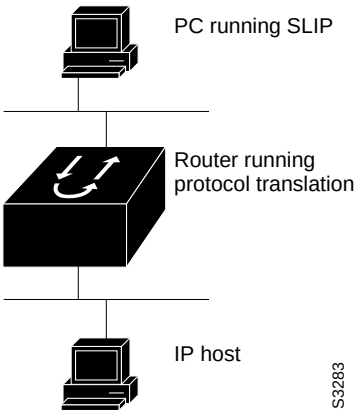
```
translate x25 31370054065 ppp 1.0.0.4 routing authentication chap
```

This is only a partial example. The commands in this example would be only part of the complete configuration file for an individual device.

Tunneling SLIP in TCP Example

A router running protocol translation software can translate between TCP and SLIP traffic to allow communication among resources in these protocol environments. In Figure 6-3, the PC running SLIP is connecting to a TCP/IP network and making a connection with the device IP host. This example enables routing and turns on header compression.

Figure 6-4 Tunneling SLIP in TCP Example



The following configuration tunnels SLIP inside of TCP packets from the SLIP client with IP address 2.0.0.5 to the router. The router then establishes a protocol translation session to IP host. Routing and header compression are enabled for the SLIP session.

```
translate tcp 1.0.0.1 slip 2.0.0.5 routing header-compression passive
```

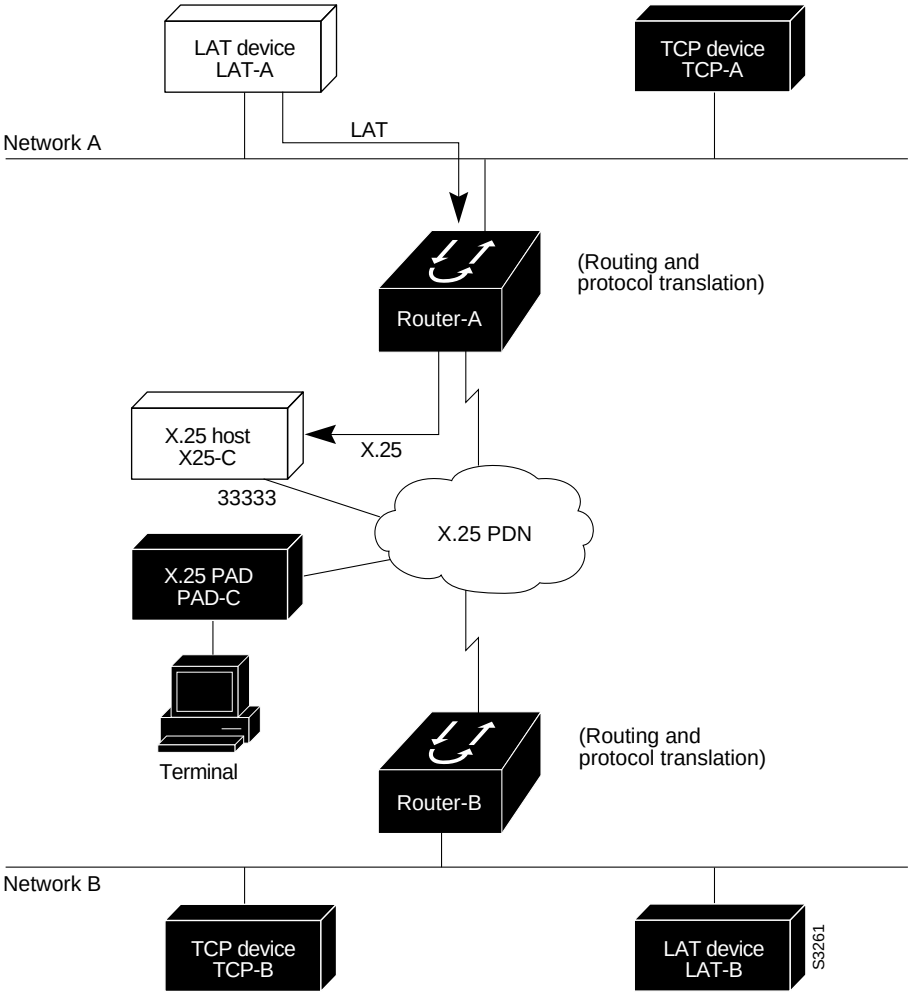
The device IP host on a different network attached to the router can be accessed by the SLIP client because routing has been enabled on the interface in the router where the SLIP session is established.

This is only a partial example. The commands in this example would be only part of the complete configuration file for an individual device.

LAT-to-X.25 Host Example

Protocol translation permits LAT devices to communicate with X.25 hosts through an X.25 PDN. In the application illustrated in Figure 6-5, 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 6-5 LAT-to-X.25 Host Translation



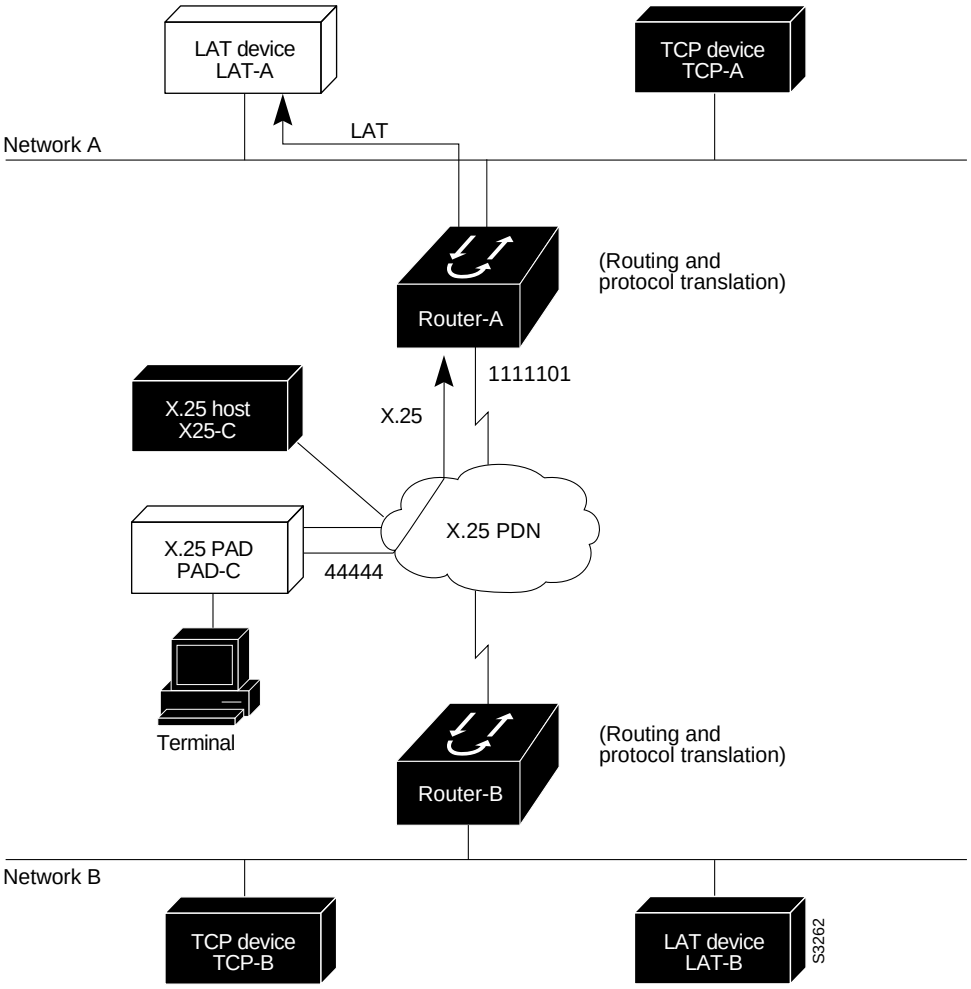
The following example illustrates how to use the **translate** global configuration command to translate from LAT to X.25. It is applied to Router-A. This example sets up reverse charging for connections, which causes the router running protocol translation software to instruct the PDN to charge the destination for the connection. It is essentially a collect call. The reversal of charges must be pre-arranged 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
```

X.25 PAD-to-LAT Example

Protocol translation permits terminals connected to X.25 PADs to communicate with LAT devices on a remote LAN (Figure 6-6). 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 router also supports X.29 access lists, which allow you to restrict LAN resources (LAT or TCP) available to the PAD user.

Figure 6-6 X.25 PAD-to-LAT Translation



The following example illustrates 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 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" later in 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.

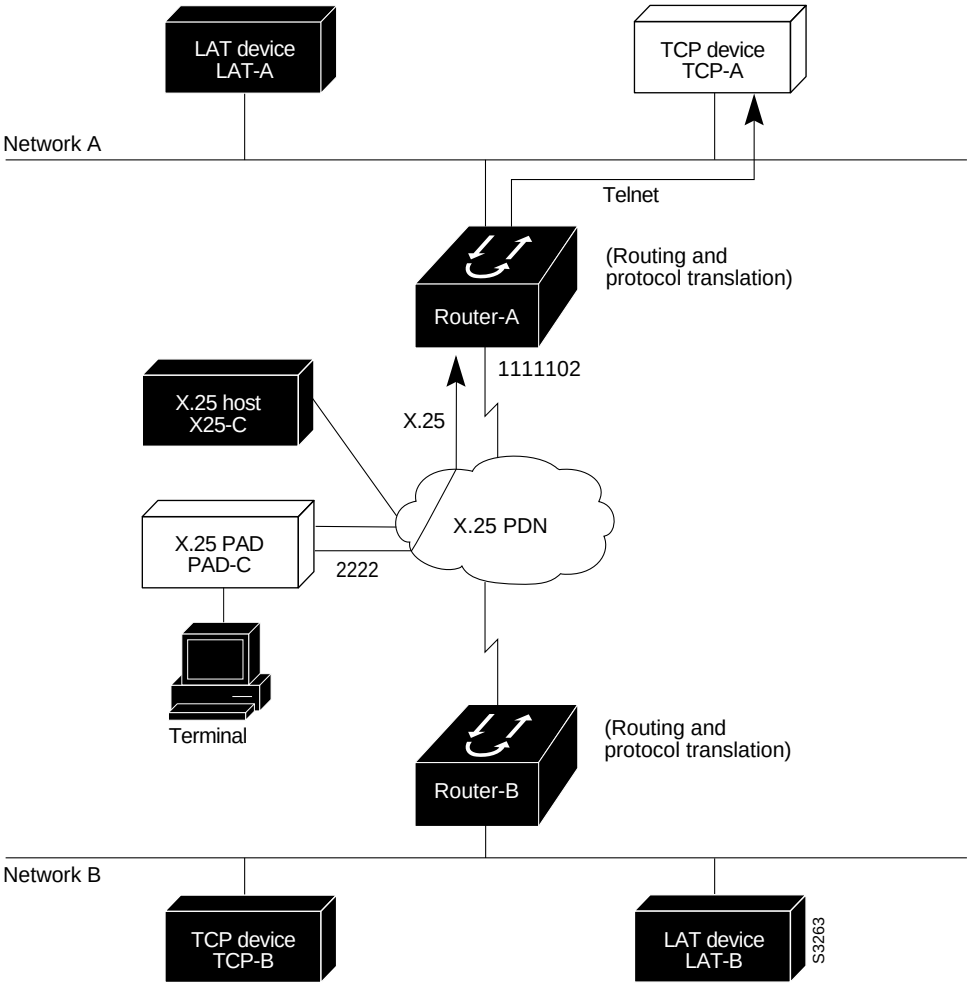
If you define multiple X.25 translate commands, each must contain a unique X.121 address. It is also important that the ITU-T protocol used in transferring packets match among X.121 addresses. This is specified in the protocol identification field of call-user data. This field specifies whether a packet 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, that 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 might or might not be the number (or a subaddress of the number) administratively assigned to the router running protocol translation software. It is up to you and the PDN to agree on a number to be used, because it is possible that the PDN can be configured to place calls on a given line that are intended for a destination 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 Example

Making a translated connection from an X.25 PAD to a TCP device (Figure 6-7) is analogous to the preceding X.25 PAD-to-LAT example. Instead of translating to LAT, the Router-A configuration includes a statement to translate to TCP (Telnet). Note that a router running protocol translation software can include statements supporting both translations (X.25 PAD-to-LAT and X.25 PAD-to-TCP). Different users on the same PAD can talk to X.25, LAT, or TCP devices.

Figure 6-7 X.25 PAD-to-TCP Translation



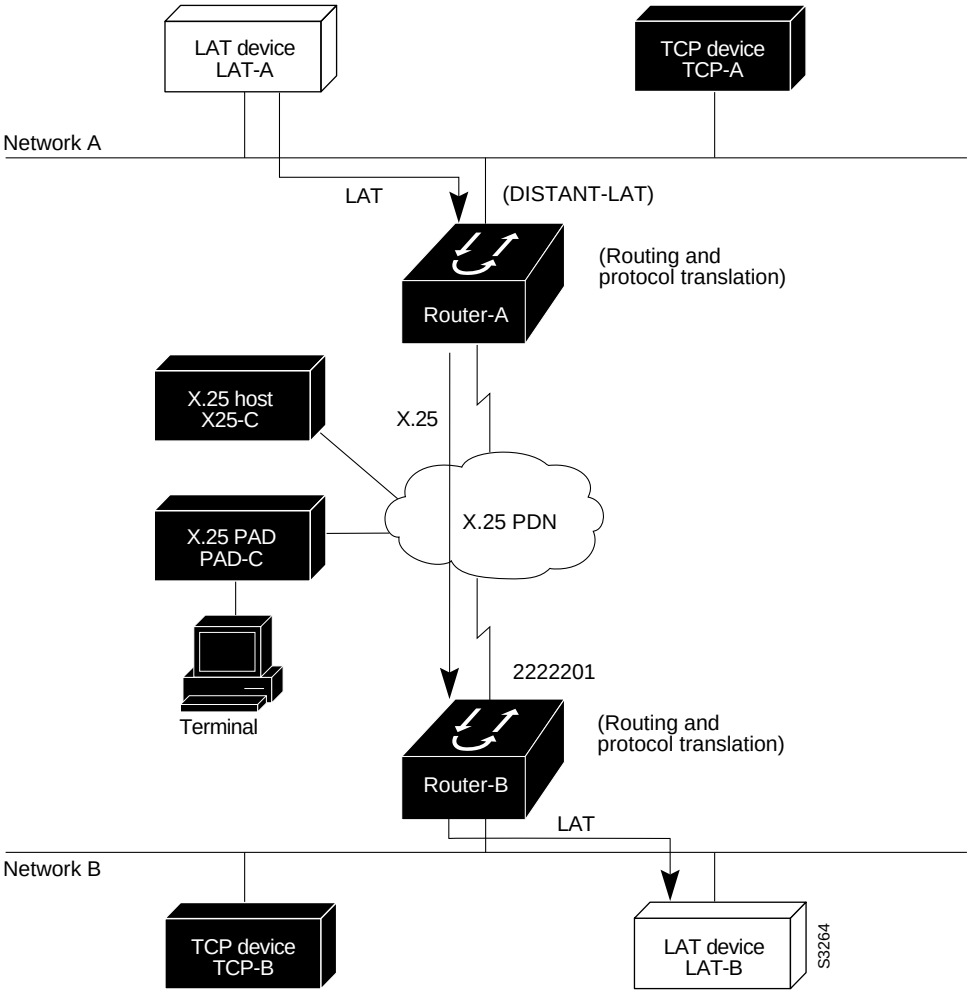
The following example illustrates how to use the **translate** global configuration command to translate from an X.25 PAD to a TCP device on Network A. It is applied to Router-A.

```
! Set up translation
translate x25 2222 tcp TCP-A
```

LAT-to-LAT via X.25 Translation Example

The protocol translation software provides transparent connectivity between LAT devices on different networks via an X.25 PDN. In Figure 6-8, which illustrates this application, the LAT device on Network A (LAT-A) first makes a virtual connection to 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 protocol translation software is configured for a one-step translation.

Figure 6-8 LAT-to-LAT via an X.25 PDN



The following two examples illustrate 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 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 TCP-B, the target specified in the **connect** command is DISTANT-LAT. (The **connect** command is described in the *Cisco Access Connection Guide*.)

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 is an example of a connection request. In this configuration example, when the user enters this command, a connection attempt from LAT-A on Network A to TCP-B on Network B is attempted.

```
local> 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 6-8):

```
! 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 running protocol translation software 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.

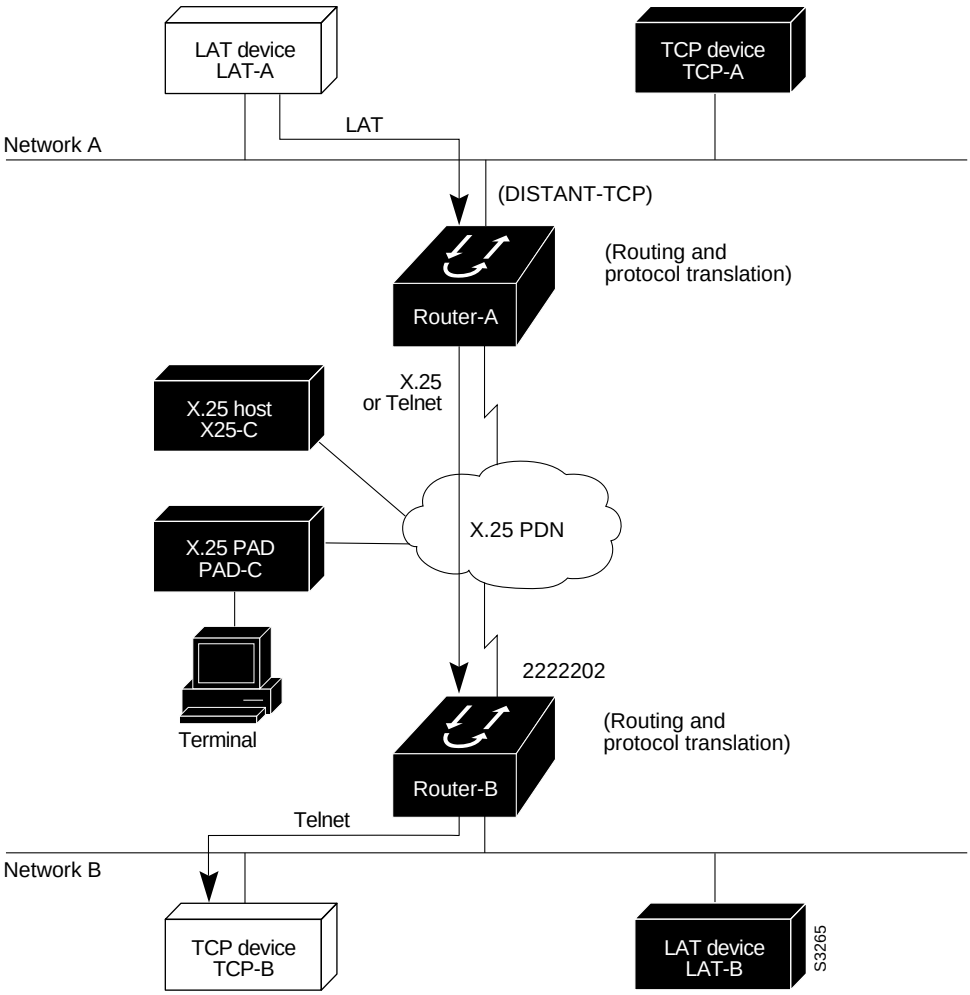
LAT-to-TCP via X.25 Example

You can use protocol translation software to provide transparent connectivity between LAT and TCP devices on different networks via an X.25 PDN. In the application illustrated in Figure 6-9, 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 to be sent out on the X.25 PDN.

If the traffic is translated from LAT directly into X.25 frames by Router-A, then Router-B on Network B translates incoming packets intended for device TCP-B into TCP. If Router-A converts LAT to TCP, then 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, while TCP over X.25 has 45 bytes of header information.

Figure 6-9 LAT-to-TCP via X.25



The following examples illustrate 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 (TCP-B) 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 is an example of a connection request. In this configuration example, 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
```

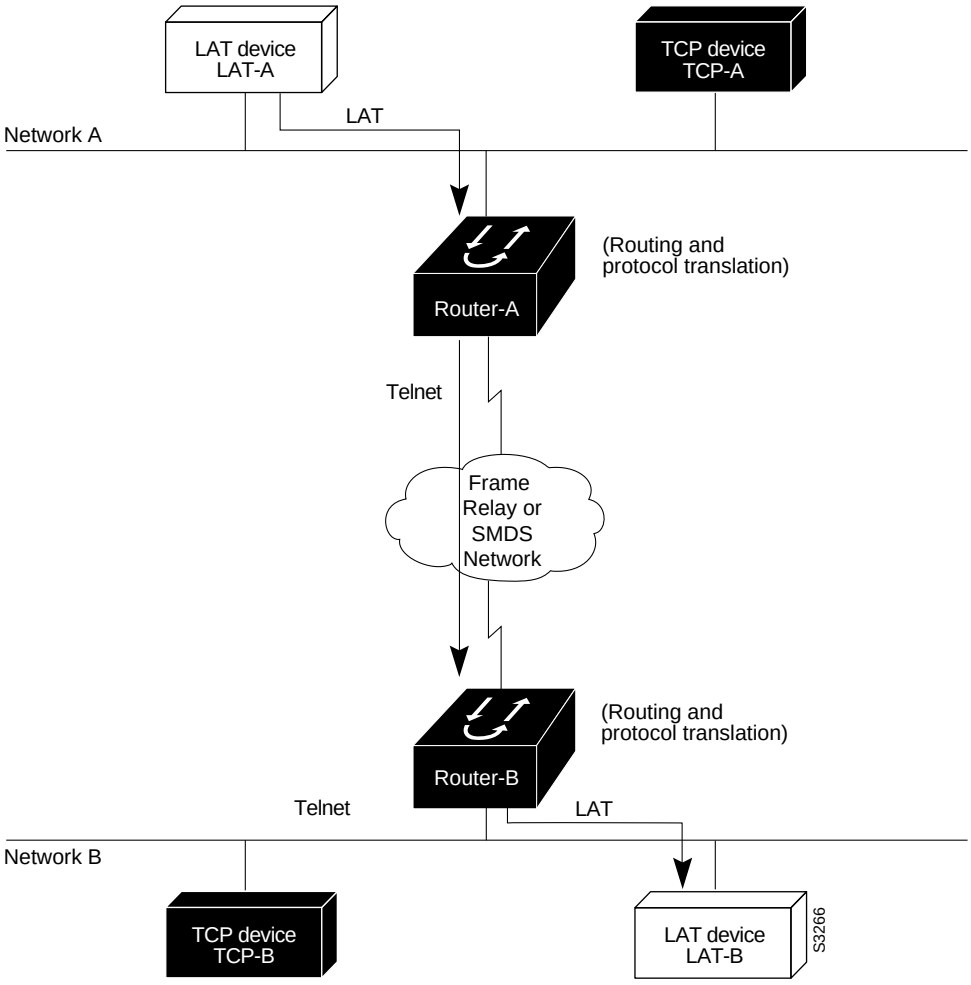
Note You can use the same name (for example, "TCP-B") in the **translate** command for both Router-A and Router-B, because each device operates independently. However, this symmetry is not required. The key is the common X.121 address used in both **translate** commands. If you prefer to have unique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

LAT-to-LAT over Frame Relay or SMDS Example

To transport LAT traffic over a Frame Relay or an 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 6-10.

Note The interface configurations for a Frame Relay or an SMDS implementation differ from the specifications at the beginning of this chapter. For more information about configuring Frame Relay and SMDS, see the *Router Products Configuration Guide*.

Figure 6-10 LAT-to-LAT over Frame Relay or SMDS



The following example illustrates how to use **translate** global configuration command to translate from LAT to LAT when the WAN uses Frame Relay or SMDS. In this configuration, the router 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
```

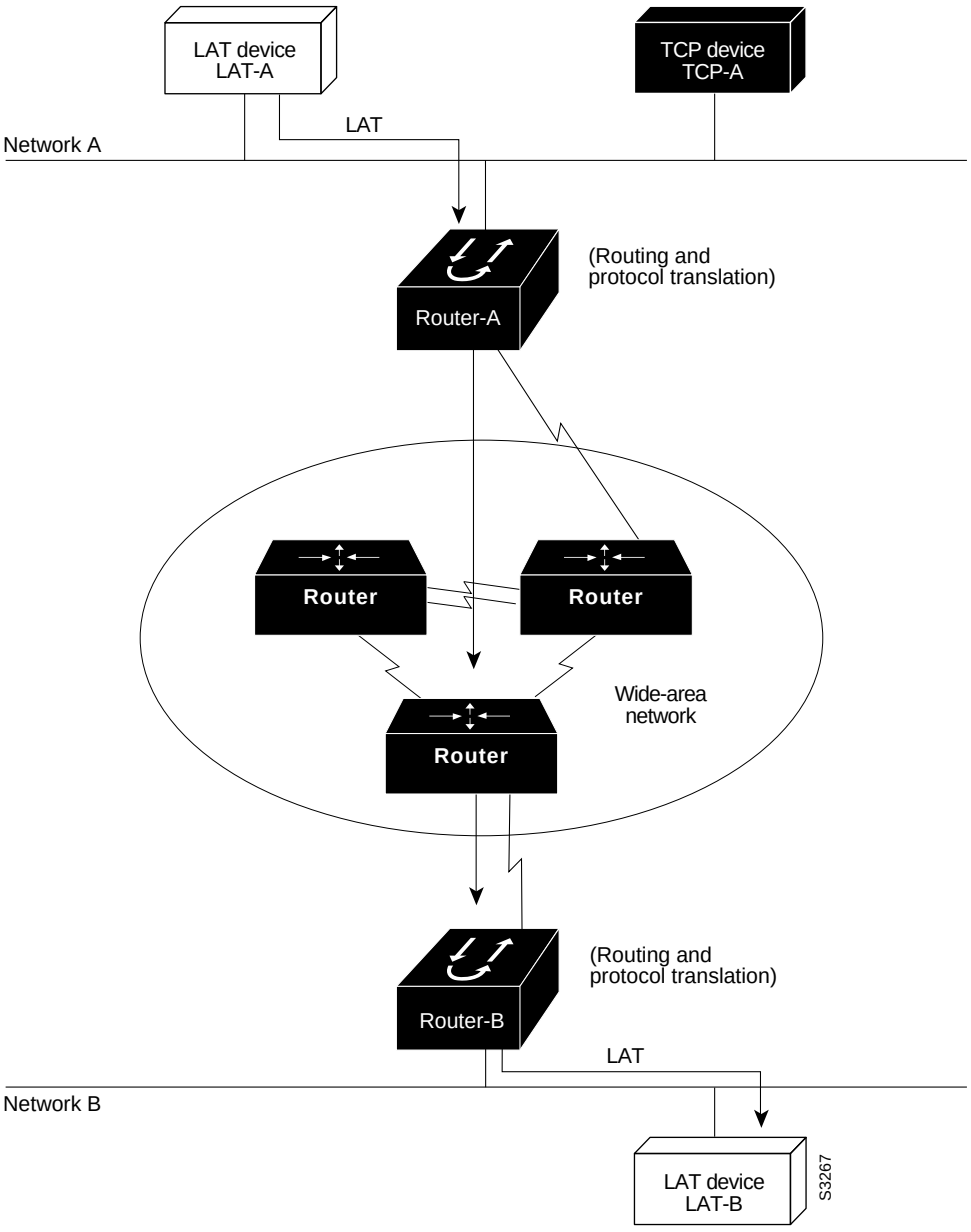
Note You can use the same name (for example, "LAT-B") in the **translate** command for both Router-A and Router-B because each device operates independently. However, this symmetry is not required. The key is the common IP name used in both **translate** commands.

LAT-to-LAT over an IP WAN Example

The protocol translation software can be used to connect LAT devices over a WAN backbone that only allows routable protocols (see Figure 6-11). This configuration exists when LAT networks are either isolated or on their own internetwork.

With the protocol translation software, 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 protocol translation software. Thus, connections are not susceptible to delays on the WAN. This 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 6-11 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, the router running protocol translation software routes encapsulated packets translated from LAT to TCP over the WAN. Packets are then translated 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 specifics of configuration for devices in the WAN. These examples are essentially the same configurations for protocol translation as those in the earlier 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
```

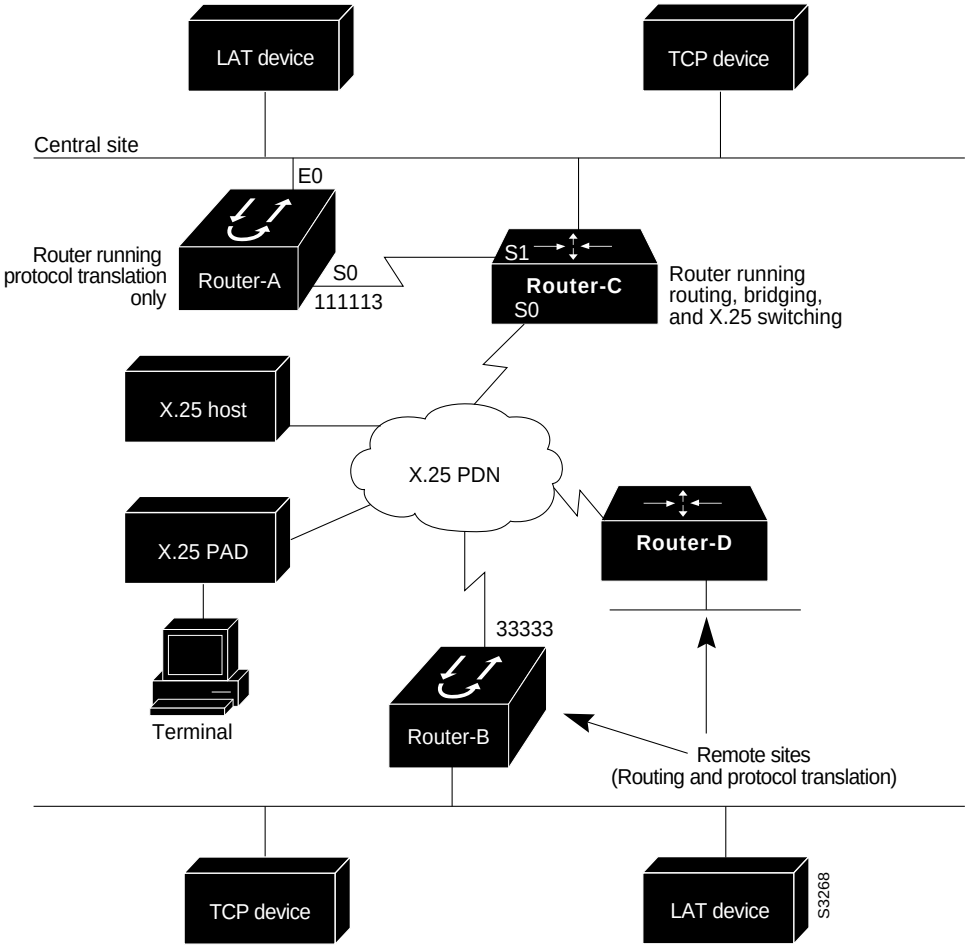
Note You can use the same name (for example, "LAT-B") in the **translate** command for both Router-A and Router-B, because each device operates independently. However, this symmetry is not required. The key is the common IP name in both **translate** commands.

Central Site Protocol Translation Example

The protocol translation software in this configuration allows more than 64 concurrent translation sessions.

To support this application, a router running protocol translation software is directly connected back-to-back (Figure 6-12) to another Cisco device. 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 6-12 Central Site Protocol Translation Example



The following example illustrates how to configure a device 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 illustrates 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 1.0.0.2 255.255.0.0
!
! 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
```

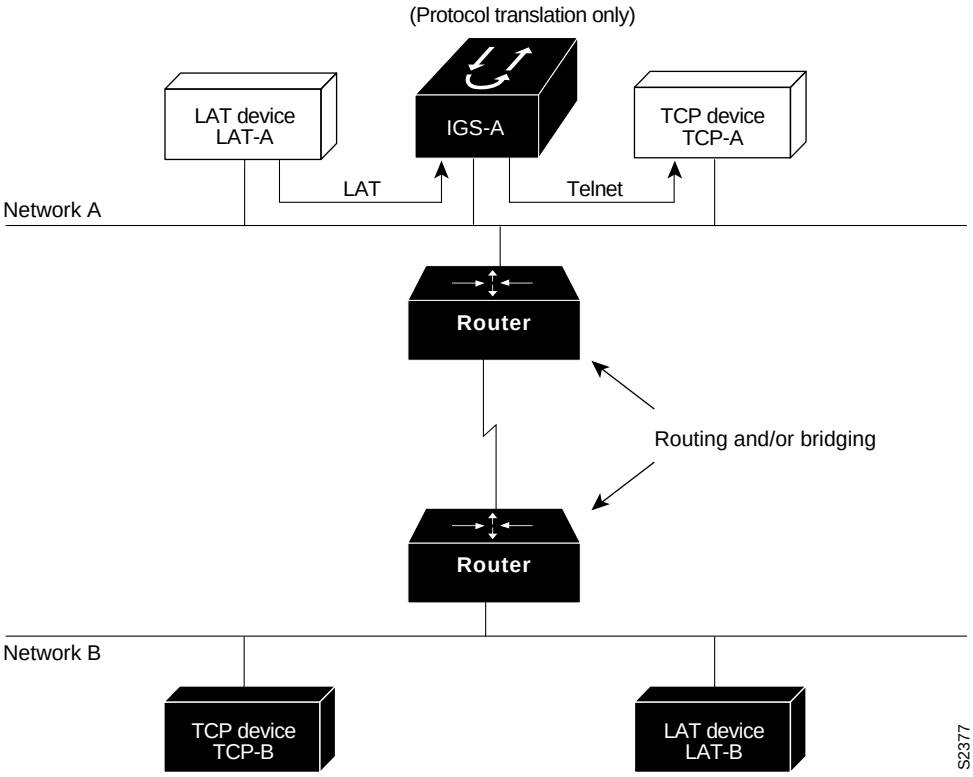
```
! “Other” Central Site Cisco Router Configuration
!
! Interface to WAN
interface serial 0
x25 address 11111
x25 route ^111113 interface serial 1
ip address ....
! Interface to Router-A
interface serial 1
x25 route .* interface serial 0
no ip address

! Translate Configuration for Router-A
!
no ip routing
! Note subaddress of subaddress 11111(3(3))
translate x25 1111133 tcp tcpdevice
translate lat TCP-B x25 3333301
translate lat tcp-device tcp tcp-device
! etc...any translate commands needed by application
```

Standalone LAT-to-TCP Translation Example

If you need a large number of local LAT-to-TCP translation sessions, you can set up Router-A to use only an Ethernet port. This application allows 100 concurrent translation sessions. In the applications illustrated in Figure 6-13, any other router that supports protocol translation can be used to interconnect network segments performing bridging or routing.

Figure 6-13 Router Functioning as a Standalone Protocol Translator




```
! Translation Configuration for Router-A only
!
interface ethernet 0
ip address 1.0.0.2 255.255.0.0
!
! enable LAT on this interface
lat enabled
!
interface serial 0
shutdown
no ip routing
default-gateway 1.0.0.100
!
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
```

X.29 Access List Example

The following example illustrates 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 131.108
access-list 2 permit 131.108.0.0 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 131.108.1.26 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. 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 131.108.1.26 x25 55551234 profile linemode
```

The X.3 PAD parameters are described in the appendix “X.3 PAD Parameters” later in this publication, and in the *Cisco Access Connection Guide*.

PART 2

Protocol Translation Command Reference

CHAPTER 7

LAT Configuration Commands

The Digital Equipment Corporation (Digital) Local Area Transport (LAT) protocol is the one used most often to connect to Digital hosts. LAT is a Digital-proprietary protocol. Cisco provides LAT technology licensed from Digital. This chapter describes the commands used to configure the LAT transmission protocol on the routers. For configuration information and examples, refer to the chapter "Configuring LAT" earlier in this publication.

access-class

To define restrictions on incoming and outgoing connections, use the **access-class** line configuration command. To remove the access-list number, use the **no** form of this command.

access-class *number* {**in** | **out**}
no access-class *number*

Syntax Description

access-list-number	Specifies an integer between 1 and 199 that defines the access list.
in	Controls which nodes can make LAT connections into the server.
out	Defines the access checks made on outgoing connections. (A user who enters a node name at the system prompt to initiate a LAT connection is making an outgoing connection.)

Default

Disabled

Command Mode

Line configuration

Usage Guidelines

This command defines access-list numbers that will then be used with the **lat access-list** command to specify the access conditions.

The value supplied for the *number* argument in both variations of the **access-class** command is used for all protocols supported by the router. If you are already using an IP access list, you must define LAT (and possibly X.25) access lists permitting connections to everything, to emulate the behavior of previous software versions.

When both IP and LAT connections are allowed from a terminal line and an IP access list is applied to that line with the **access-class** command, you must also create a LAT access list with the same number if you want to allow any LAT connections from that terminal. You can specify only one incoming and one outgoing access list number for each terminal line. When checking LAT access lists, if the specified list does not exist, the system denies all LAT connections.

Example

The following example configures an incoming access class on virtual terminal line 4:

```
line vty 4
access-class 4 in
```

Related Command

lat access-list

clear entry

To delete an entry from the list of queued host-initiated connections, enter the **clear entry** EXEC command at the system prompt.

clear entry *number*

Syntax Description

number An entry number obtained from the **show entry** EXEC command.

Command Mode

EXEC

Example

The following example illustrates how to delete pending entry number 3 from the queue:

```
pt# clear entry 3
```

Related Command

show entry

lat access-list

To specify access conditions to nodes on the LAT network, use the **lat access-list** global configuration command. To remove a specified access-list number, use the **no** form of this command.

lat access-list *number* {**permit** | **deny**} *nodename*
no lat access-list *number*

Syntax Description

number	Specifies a number between 1 and 199 assigned to the line using the **access-class** line configuration command.
permit	Allows any matching node name to access the line.
deny	Denies access to any matching node name.
nodename	Specifies the name of the LAT node, with or without regular expression pattern matching characters, with which to compare for access. The UNIX-style regular expression characters allow for pattern matching of characters and character strings in the node name.

Default

No access conditions defined.

Command Mode

Global configuration

Usage Guidelines

Regular expressions are case sensitive. Because LAT node names are always in all capital letters, make sure you use only all capital-letter regular expressions.

Table 7-1 and Table 7-2 summarize pattern-matching and character-matching symbols and their use. A more complete description of the pattern matching characters is found in the appendix "Regular Expressions" later in this publication.

Table 7-1 Pattern Matching

Character	Description
\0	Replaces the entire original address.
\1..9	Replaces the strings that match the first through ninth parenthesized part of X.121 address.
*	Matches 0 or more sequences of the regular expressions.
+	Matches 1 or more sequences of the regular expressions.
?	Matches the regular expression of the null string.

Table 7-2 Character Matching

Character	Description
^	Matches the null string at the beginning of the input string.
$	Matches the null string at the end of the input string.
char	Matches *char*.
.	Matches any single character.

Examples

The following example illustrates how to enter a request to permit all packets destined for any LAT node named *WHEEL*:

```
lat access-list 1 permit WHEEL
```

The following example illustrates how to enter a request to deny all packets destined for any LAT node name beginning with the *BLDG1-* prefix:

```
lat access-list 2 deny ^BLDG1-
```

Related Command

access-class

lat enabled

To enable LAT, use the **lat enabled** interface configuration command. To disable LAT, use the **no** form of this command.

lat enabled
no lat enabled

Syntax Description

This command has no arguments or keywords.

Default

Enabled

Command Mode

Interface configuration

Examples

The following example enables LAT on the Ethernet 0 interface:

```
interface ethernet 0
lat enabled
```

The following example disables LAT on the same Ethernet interface:

```
interface ethernet 0
no lat enabled
```

lat group-list

Use the **lat group-list** global configuration command to allow a name to be assigned to the group list. A group list is any combination of group names, numbers, or ranges. To remove the specified group list, use the **no** form of this command.

lat group-list *groupname* {*number* | *range* | **all**} [**enabled | disabled**]
no lat group-list *groupname* {*number* | *range* | **all**} [**enabled | disabled**]

Syntax Description

groupname	Specifies a group code name.
number	Specifies a group code number. You can also enter both a group code name and group code numbers.
range	Specifies a hyphenated range of numbers.
all	Specifies the range from 0 to 255.
enabled	(Optional) Allows incremental changes to the list; that is, you can add a group code without retyping the entire command.
disabled	(Optional) Allows selective removal of a group code from the list.

Default

No group names are assigned to the list.

Command Mode

Global configuration

Usage Guidelines

Specifying a name for a group list simplifies the task of entering individual group codes. In other words, a name makes it easier to refer to a long list of group code numbers. The group list must already exist. Use the EXEC command **show lat groups** to see a list of existing groups.

Examples

The following example creates the new group named *stockroom* and defines it to include the group numbers 71 and 99:

```
lat group-list stockroom 71 99
```

The following example adds group code 101 to the group named *stockroom*:

```
lat group-list stockroom 101 enabled
```

The following example deletes the group named *Bldg-2*:

```
no lat group-list Bldg-2
```

Related Commands

lat out-group
lat service-group

lat host-buffers

To set the number of receive buffers that will be negotiated when the router is acting as a LAT host, use the **lat host-buffers** global configuration command. To return to the default of one receive buffer, use the **no** form of this command.

lat host-buffers *receive-buffers*
no lat host-buffers *receive-buffers*

Syntax Description

receive-buffers	An integer that specifies the number of receive buffers that will be negotiated. The argument can be any number between 1 and 128.

Default

1 receive buffer

Command Mode

Global configuration

Usage Guidelines

Before LAT Version 5.2, LAT allowed only one outstanding message at a time on a virtual circuit. This could limit the performance of large routers. For example, only one Ethernet packet of data could be in transit at a time. With LAT Version 5.2, nodes can indicate that they are willing to receive more than one message at a time. During virtual-circuit startup, each side communicates to the other how many outstanding messages it is willing to accept.

Example

The following example enables LAT and configures the LAT host to negotiate 100 receive buffers:

```
lat enabled
lat host-buffers 100
```

Related Command

lat server-buffers

lat ka-timer

To set the rate of the keepalive timer, use the **lat ka-timer** global configuration command. To restore the default, use the **no** form of this command.

lat ka-timer *seconds*
no lat ka-timer

Syntax Description

seconds Timer rate in seconds.

Default

20 seconds

Command Mode

Global configuration

Usage Guidelines

The keepalive timer sets the rate that messages are sent in the absence of actual traffic between the router and the remote node. The server uses keepalive messages to detect when communication with a remote node is disrupted or when the remote node has crashed.

Example

The following example sets the keepalive timer to rate of five seconds:

```
lat ka-timer 5
```

lat node

To change the LAT node name without changing the system host name, use the **lat node** global configuration command.

lat node *node-name*

Syntax Description

node-name Name of the LAT node.

Default

No default LAT node name

Command Mode

Global configuration

Usage Guidelines

This command allows you to give the server a node name that is different than the host name. Use the EXEC command **show entry** to determine which LAT hosts have queue entries for printers on the servers. Use the EXEC command **clear entry** to delete entries from the queue.

Example

The following example specifies the LAT node name as *DEC2*:

```
lat node DEC2
```

Related Commands

clear entry
show entry

lat out-group

To define a group list for a line's outgoing user-initiated connections, use the **lat out-group** line configuration command.

lat out-group {*groupname* | *number* | *range* | **all**}

Syntax Description

groupname	Specifies a group code name.
number	Specifies a group code number.You can also enter both a group code name and group code numbers.
range	Specifies a hyphenated range of numbers.
all	Specifies the range from 0 to 255.

Default

The default group code number is 0.

Command Mode

Line configuration

Usage Guidelines

Use the EXEC command **show lat** to display group numbers. If the host node and router do not share a common group number, the host's services will not be displayed.

Use the command **lat out-group 0** to return to the default value of 0.

Example

The following example defines the services for lines 1 through 7, 10 through 17, and 20 through 24. Access to systems on the first set of lines is limited to groups 12 and 18 through 23; the second set is limited to group 12; the third set is limited to group codes 12, 18 through 23, and 44. All other lines use the default of group zero.

```
line 1 7
lat out-group 12 18-23
line 10 17
lat out-group 12
line 20 24
lat out-group 12 18-23 44
```

Related Commands

Two daggers (††) indicate that the command is documented in the *Cisco Access Connection Guide*.

terminal lat out-group††
lat group-list

lat retransmit-limit

To set the number of times that LAT retransmits a message before declaring the remote system unreachable, use the **lat retransmit-limit** global configuration command. To restore the default retry value, use the **no** form of this command.

lat retransmit-limit *number*
no lat retransmit-limit

Syntax Description

number	Number of retries—any number between 4 and 255.

Default

8 retries

Command Mode

Global configuration

Usage Guidelines

Assigning larger values to the number of tries increases the robustness of the LAT service at the cost of longer delays when communications are disrupted. Because LAT generally retransmits messages once a second, the value is approximately the number of seconds that LAT connections will survive connection disruption.

If you bridge LAT, the message-retransmit limit should be set to at least 20 tries for LAT sessions to survive a worst-case spanning-tree reconfiguration, because bridging spanning-tree reconfiguration can take up to 15 seconds.

Example

The following example sets the retransmission limit to 30 tries, enough time to sustain the down time incurred when the system must reconfigure a spanning-tree topology:

```
lat retransmit-limit 30
```

lat server-buffers

To set the number of receive buffers that will be negotiated when the router is acting as a LAT server, use the **lat server-buffers** global configuration command. To return to the default of one receive buffer, use the **no** form of this command.

lat server-buffers *receive-buffers*
no lat server-buffers *receive-buffers*

Syntax Description

receive-buffers	Integer that specifies the number of receive buffers that will be negotiated. The argument can be any number between 1 and 128.

Default

1 receive buffer

Command Mode

Global configuration

Usage Guidelines

Before LAT Version 5.2, LAT allowed only one outstanding message on a virtual circuit at a time. This could limit the performance of large routers because only one Ethernet packet of data could be in transit at a time. With LAT Version 5.2, nodes can indicate that they are willing to receive more than one message at a time. During virtual-circuit startup, each side communicates to the other how many outstanding messages it is willing to accept.

Example

The following example enables LAT and configures the server to negotiate 25 receive buffers:

```
lat enabled
lat server-buffers 25
```

Related Command

lat host-buffers

lat service autocommand

To associate a command with a service, use the **lat service autocommand** global configuration command. To remove the specified autocommand, use the **no** form of this command.

lat service *service-name* **autocommand** *command*
no lat service *service-name* **autocommand** *command*

Syntax Description

service-name	Name of the service.
command	Command to be associated with the service.

Default

No commands automatically associated with a service.

Command Mode

Global configuration

Usage Guidelines

When an inbound connection is received for the specified service, the command associated with the service is automatically executed instead of the user receiving a virtual terminal session.

TACACS or port passwords are bypassed for these services; only the LAT password is checked.

Note Do not use this option with the **rotary** keyword.

Example

The following example associates the command **telnet sartre** to the service *SARTRE*:

```
lat service SARTRE autocommand telnet sartre
```

lat service enabled

To enable inbound connections to the specified service and enable the advertisement of this service to routers on the network, use the **lat service enabled** global configuration command. To delete the named service, use the **no** form of this command.

lat service *service-name* **enabled**
no lat service *service-name* **enabled**

Syntax Description

service-name	Name of the service.

Default

No services enabled

Command Mode

Global configuration

Usage Guidelines

In the simplest form, this command creates a service that gives connecting users access to a VTY port on the server.

Use the **enabled** keyword after commands that define a service so that users do not connect to a service before all the parameters are set.

Deleting a service does not disconnect existing connections.

Example

The following example illustrates how to enable inbound connections to the service *WHEEL*:

```
lat service WHEEL enabled
```

lat service ident

To set the LAT service identification for a specified service, use the **lat service ident** global configuration command. To remove the identification, use the **no** form of this command.

lat service *service-name* **ident** *identification*
no lat service *service-name* **ident**

Syntax Description

service-name	Name of the service.
identification	Descriptive name (text only) that identifies the service.

Default

No LAT service identification set for specific services.

Command Mode

Global configuration

Usage Guidelines

The identification is advertised to other servers on the network and is displayed along with the list of name services on the LAN.

Example

The following example specifies the identification *Welcome to Gateway-A* on service *STELLA*:

```
lat service STELLA ident Welcome to Gateway-A
```

lat service password

To set up a LAT password for a service, use the **lat service password** global configuration command. To remove the password, use the **no** form of this command.

lat service *service-name* **password** *password*
no lat service *service-name* **password**

Syntax Description

service-name	Name of the service.
password	Password.

Default

No default LAT service passwords

Command Mode

Global configuration

Usage Guidelines

The connecting user will be required to enter the password to complete the connection.

The password is obtained through the LAT password mechanism; routers running Software Release 8.1 or earlier do not support this capability. Any services protected in this manner cannot be connected by a device running 8.1 or earlier software.

Example

The following example specifies a service named *BLUE* and the password *secret*:

```
lat service BLUE password secret
```

lat service rating

To set a static service rating for the specified service, use the **lat service rating** global configuration command. To remove the service rating, use the **no** form of this command.

lat service *service-name* **rating** *static-rating*
no lat service *service-name* **rating**

Syntax Description

service-name	Name of the service.
static-rating	Static service rating. The rating must be in the range of 1 to 255.

Default

Dynamic rating

Command Mode

Global configuration

Usage Guidelines

If this command is not entered, the router calculates a dynamic rating based on the number of free ports that can handle connections to the service. Setting a static rating overrides this calculation and causes the specified value to be used.

Example

The following example specifies a service rating of 84 on the service *WHEEL*:

```
lat service WHEEL rating 84
```

lat service rotary

To associate a rotary group with a service, use the **lat service rotary** global configuration command. To remove the association, use the **no** form of this command.

lat service *service-name* **rotary** *group*
no lat service *service-name* **rotary**

Syntax Description

service-name	Name of the service.
group	Rotary group number.

Default

Disabled

Command Mode

Global configuration

Usage Guidelines

Establish rotary groups using line configuration commands and the **rotary** line configuration command.

When an inbound connection is received for this service, the router establishes a reverse-LAT connection to a terminal in that rotary group.

If the rotary option is not set, the connection will be to a virtual terminal session on the router.

Example

The following example creates a service called *MODEM* to establish a rotary group:

```
lat services MODEM rotary 1
```

Related Command

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

rotary†

lat service-announcements

To reenable LAT broadcast service announcements, use the **lat service-announcements** global configuration command. To disable the sending of LAT service announcements, use the **no** form of this command.

lat service-announcements
no lat service-announcements

Syntax Description

This command has no arguments or keywords.

Default

Enabled

Command Mode

Global configuration

Usage Guidelines

If this command is enabled, the LAT code will periodically broadcast service advertisements. If the command is disabled, the LAT code will not send service announcements, so solicit information messages will have to be used to look up node information.

Note You should only disable service announcements if all of the nodes on the local-area network (LAN) support the service responder feature.

Example

The following example reenables the sending of broadcast service announcements:

```
lat service-announcements
```

Related Command

lat service-responder

lat service-group

To specify a group code mask to use when advertising all services for this node and to control incoming services, use the **lat service-group** global configuration command. To remove the group code mask specified, use the **no** form of this command.

lat service-group {*groupname* | *number* | *range* | **all**} [**enabled | disabled**]
no lat service-group {*groupname* | *number* | *range* | **all**} [**enabled | disabled**]

Syntax Description

groupname	Specifies a group code name.
number	Specifies a group code number.
range	Specifies a hyphenated range of numbers between 0 and 255.
all	Specifies the range from 0 to 255.
enabled	(Optional) Allows incremental changes to the list; that is, you can add a group code without retyping the entire command.
disabled	(Optional) Allows selective removal of a group code from the list.

Default

If no service group is specified, the router defaults to advertising to group 0.

Command Mode

Global configuration

Usage Guidelines

When this command is written to nonvolatile memory (using the EXEC **write memory** command), the system looks for an exact match on a group code name. If it finds one, it uses that name in the command. Otherwise, it writes out a list of numbers, using the range syntax whenever possible.

Examples

The following example specifies groups 100 through 103, then defines engineering as the group code list to advertise:

```
lat group-list engineering 100-103
lat service-group engineering enabled
```

The following example specifies the groups 1, 5, 20 through 36, and 52:

```
lat service-group 1 5 20-36 52
```

You can then enter the following command to add group 99:

```
lat service-group 99 enabled
```

Related Command

lat group-list

lat service-responder

To configure a node to act as proxy for other nodes when a solicit-information multicast message is received, use the **lat service-responder** global configuration command. To remove any proxy definition set up using the **lat service-responder** command, use the **no** form of this command.

lat service-responder
no lat service responder

Syntax Description

This command has no arguments or keywords.

Default

Disabled

Command Mode

Global configuration

Usage Guidelines

Your router can be configured to support the service responder feature that is part of the latest LAT Version 5.2 specification.

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

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

Your router can be configured as a LAT service responder. If all your nodes are LAT Version 5.2 nodes, you do not need to enable the service responder features.

Example

The following example configures a node to act as a proxy for a node when a solicit-information multicast message is received. The node configured with this command will respond to solicit messages.

```
lat service-responder
```

Related Command

lat service-announcements

lat service-timer

To adjust the time between LAT service advertisements, use the **lat service-timer** global configuration command.

lat service-timer *interval*

Syntax Description

interval Number of seconds between service announcements. Note that the granularity offered by this command is ten-second intervals, and the *interval* value is rounded up.

Default

20 seconds

Command Mode

Global configuration

Usage Guidelines

This command adjusts the time, in seconds, between LAT service announcements for services offered by the router. This is useful in large networks with many LAT services and limited bandwidth.

Example

The following example sets the interval between LAT service announcements to 11, and illustrates the rough granularity of the **lat service-timer** command:

```
! The time between LAT service announcements is set to 11. Because the
! granularity is in ten-second intervals, the actual time between announcements
! is 20 seconds.
lat service-timer 11
! 20 seconds between updates
lat service-timer 19
! 120 seconds between updates
lat service-timer 120
```

lat vc-sessions

To set the maximum number of sessions to be multiplexed onto a single LAT virtual circuit, use the **lat vc-sessions** global configuration command. To remove a prior session's definition set, use the **no** form of this command.

lat vc-sessions *number*
no lat vc-sessions *number*

Syntax Description

number	Specifies the number of sessions that will be multiplexed onto a single LAT virtual circuit. This number cannot be greater than 255.

Default

255 sessions per virtual circuit

Command Mode

Global configuration

Usage Guidelines

Setting the number of sessions to a smaller number can increase throughput if there are a large number of sessions to one host, especially with terminal servers with many physical ports. It can also increase overhead if there is little traffic but a large number of sessions to the same host

Example

The following example sets the maximum number of sessions to be multiplexed onto a single LAT virtual circuit at 100:

```
lat vc-sessions 100
```

lat vc-timer

To set the interval of time LAT waits before sending traffic, use the **lat vc-timer** global configuration command. To remove a timer definition, use the **no** form of this command.

lat vc-timer *milliseconds*
no lat vc-timer *milliseconds*

Syntax Description

milliseconds	Timer value. Specifies the amount of time LAT will wait before sending traffic. Acceptable values are between 10 and 1000 milliseconds.

Default

80 milliseconds

Command Mode

Global configuration

Usage Guidelines

Smaller timer values increase the overhead on both the router and the host. However, you can use smaller values to correct buffer overflows, which happen when the router receives more data than it can buffer during a virtual-circuit timer interval.

Larger values increase the need for router buffering and can cause noticeable echoing delay. However, increased values can reduce traffic. In environments with slow bridging, retransmissions can be reduced if you increase the value to at least three times the worst-case, round-trip interval.

Example

The following example sets the time between transmitting messages to 500 milliseconds:

```
lat vc-timer 500
```

show entry

Use the **show entry** EXEC command to display the list of queued host-initiated connections to a router. You can use this command to determine which LAT hosts have queue entries for printers on routers.

show entry

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Sample Display

The following is sample output from the **show entry** command. The display shows that two LAT connections are waiting for access to port 5. The list is ordered so that the lower-numbered entry has been waiting longer, and will get to use the line next.

```
sloth# show entry

1 waiting 0:02:22 for port 5 from LAT node BLUE
2 waiting 0:00:32 for port 5 from LAT node STELLA
```

Table 7-3 describes the fields in the first line of output shown in the display.

Table 7-3 Show Entry Field Descriptions

Field	Description
1	Number assigned to the queued connection attempt
waiting 0:02:22	Interval (hours:minutes:seconds) during which the connection attempt has been waiting
for port 5	Port for which the connection attempt is waiting
from LAT node BLUE	Name of the user (BLUE) attempting to make the connection

show lat advertised

Use the **show lat advertised** EXEC command to display the LAT services a router offers to other systems running LAT on the network.

show lat advertised

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Usage Guidelines

Advertised services are created with the **lat service** configuration commands. The display includes the service rating, rotary group, if present, and whether or not the service is enabled for incoming connections.

Sample Display

The following is sample output from the **show lat advertised** command:

```
sloth# show lat advertised

Service Name         Rating     Rotary  Flags
SARTRE              4(Dynamic)   None  Enabled
  Autocommand: telnet sartre
MODEMS              0(Dynamic)     12  Enabled
  Ident: SpaceBlazer modem services
RECLUSE             4(Dynamic)   None  Enabled
  Ident: white recluse...
```

The display shows output from a router, *RECLUSE*, which has three services defined: *SARTRE*, *MODEMS*, and *RECLUSE*.

Table 7-4 describes significant fields shown in the display.

Table 7-4 Show LAT Advertised Field Descriptions

Field	Description
Service Name	Lists the LAT service name.
Rating	Lists the static service rating set, if any.
Rotary	Lists the associated rotary service.
Flags	Lists whether or not a service is enabled.
Autocommand	Defines the autocommand associated with the service.
Ident	Lists the advertised identification for the service.

show lat groups

Use the **show lat groups** EXEC command to display the groups that were defined for a router using the **lat group-list** global configuration command.

show lat groups

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Sample Display

The following is sample output from the **show lat groups** command:

```
sloth# show lat groups

Group Name          Len    Groups
cafeteria           3      13   15   23
engineering         7      55
manufacturing       10     70   71   72
```

Table 7-5 describes significant fields shown in the display.

Table 7-5 Show LAT Groups Field Descriptions

Field	Description
Group Name	Assigned group name
Len	Size of internal data structure used to contain the group code map
Groups	Group codes associated with the learned group

Related Command

lat group-list

show lat nodes

Use the **show lat nodes** EXEC command on a router to display information about all known LAT nodes.

show lat nodes

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Sample Display

The following is sample output from the **show lat nodes** command:

```
pt# show lat nodes

Node "CHAOS", usage -1, Interface Ethernet0, Address 0000.0c01.0509
  Timer 89,  sequence 188,  changes 131,  flags 0x0, protocol 5.1
  Facility 0,  Product code 0,  Product version 0
  Recv 0/0/0,  Xmit 0/0/0,  0 Dups, 0 ReXmit
  Bad messages: 0,  Bad slots: 0,  Solicits accepted: 0
  Solicits rejected: 0,  Multiple nodes: 0
  Groups:   0
  Service classes:   1
Node "CONFUSED", usage -1, Local
  Timer 99,  sequence 4,  changes 151,  flags 0x0, protocol 5.2
  Facility 0,  Product code 0,  Product version 0
  Recv 0/0/0,  Xmit 0/0/0,  0 Dups, 0 ReXmit
  Bad messages: 0,  Bad slots: 0,  Solicits accepted: 0
  Solicits rejected: 0,  Multiple nodes: 0
  Groups:   0
  Service classes:   1
Node "WOMBAT", usage -1, Interface Ethernet0, Address 0000.0cff.c9ed
  Timer 99,  sequence 9,  changes 159,  flags 0x0, protocol 5.1
  Facility 0,  Product code 0,  Product version 0
  Recv 0/0/0,  Xmit 0/0/0,  0 Dups, 0 ReXmit
  Bad messages: 0,  Bad slots: 0,  Solicits accepted: 0
  Solicits rejected: 0,  Multiple nodes: 0
  Groups:   0
  Service classes:   1
Node "TARMAC", usage -1, Interface Ethernet0, Address 0000.0c02.c7c1
  Timer -10351,  sequence 1,  changes 131,  flags 0x40, protocol 5.2
  Facility 0,  Product code 0,  Product version 0
  Recv 0/0/0,  Xmit 0/0/0,  0 Dups, 0 ReXmit
  Bad messages: 0,  Bad slots: 0,  Solicits accepted: 0
  Solicits rejected: 0,  Multiple nodes: 0
  Groups:   0
  Service classes:   1
```

Table 7-6 describes significant fields shown in the display.

Table 7-6 Show LAT Nodes Field Descriptions

Field	Description
Node	The node name as reported by the host computer.
usage	The number of virtual circuits currently active to this node.
Interface	Node interface type and number.
Address	The MAC address of the node's Ethernet interface.
Timer	The number of seconds remaining until this node's service announcements message will time out; this value is set to three times the nodes multicast timer value whenever a new service announcements is received.
sequence	The sequence number received in the last service announcements message received. Nodes increment their sequence number when the contents of the service advertisement change.
changes	The internal representation of what changed in the multicast message the last time the sequence number changed.
flags	The internal representation of various state information about the node.
protocol	The LAT protocol version used by the node.
Facility	The remote facility number.
Product code	The remote product code.
Product version	The remote product version.
Recv and Xmit	The number of messages, slots, and bytes received or transmitted to the node. The number of messages is the number of LAT virtual circuit messages. Each virtual-circuit message contains some number of slots, which contain actual terminal data or control information. Bytes is the number of data bytes (input or output characters) exchanged.
Dups	The number of duplicate virtual-circuit messages received.
ReXmit	The number of virtual-circuit messages retransmitted.
Bad messages	The number of bad messages received.
Bad slots	The number of bad slots received.
Solicits accepted	The number of solicit-information requests accepted.
Solicits rejected	The number of solicit-information requests rejected.
Multiple nodes	The total of multiple nodes seen.
Groups	The list of group codes advertised by the node's service announcements.
Service classes	The number of service classes.

show lat sessions

Use the **show lat sessions** EXEC command on a router to display information on active LAT sessions.

show lat sessions [*line-number*]

Syntax Description

line-number	(Optional) Use to display information about a single line.

Command Mode

EXEC

Sample Displays

The following is sample output from the **show lat sessions** command:

```
orange# show lat sessions

tty0, connection 1 to service TERM1
TTY data:
  Name ”0”, Local usage 1/0, Remote usage disabled
  Flags: Local Connects, Enabled
  Command Mode flags: none
  Config flags: -FlowOut, -FlowIn, Parameter Info
  Flow control ^S/^Q in ^S/^Q out,  Mode Normal, Parity None, databits 8
  Groups:   0
Session data:
  Name TERM1, Remote Id 1, Local Id 1
  Remote credits 2, Local credits 0, Advertised Credits 2
  Flags: none
  Max Data Slot 255, Max Attn Slot 255, Stop Reason 0
Remote Node data:
Node ”TERM1”, Address 0000.0C00.291F, usage 1
  Timer 59,  sequence 5,  changes 159,  flags 0x0, protocol 5.1
  Recv 56/22/83,  Xmit 41/23/14,  0 Dups, 0 ReXmit
  Groups:   0
tty10, connection 1 to service ENG2
TTY data:
  Name ”10”, Local usage 1/0, Remote usage disabled
  Flags: Local Connects, Enabled
  Command Mode flags: none
  Config flags: -FlowOut, +FlowIn, Set Parameters, 0x40000000
  Flow control ^S/^Q in ^S/^Q out,  Mode Normal, Parity None, databits 8
  Groups:   0
Session data:
  Name ENG2, Remote Id 1, Local Id 1
  Remote credits 1, Local credits 0, Advertised Credits 2
  Flags: none
  Max Data Slot 255, Max Attn Slot 255, Stop Reason 0
Remote Node data:
Node ”ENG2”, Address AA00.0400.34DC, usage 1
  Timer 179,  sequence 60,  changes 255,  flags 0x0, protocol 5.1
  Recv 58/29/186,  Xmit 50/36/21,  0 Dups, 0 ReXmit
  Groups:   0
```

The following display shows information about active LAT sessions on one line, line 10:

```
pt# show lat sessions 10

tty10, connection 1 to service ENG2
TTY data:
  Name ”10”, Local usage 1/0, Remote usage disabled
  Flags: Local Connects, Enabled
  Command Mode flags: none
  Config flags: -FlowOut, +FlowIn, Set Parameters, 0x40000000
  Flow control ^S/^Q in ^S/^Q out,  Mode Normal, Parity None, databits 8
  Groups:   0
Session data:
  Name ENG2, Remote Id 1, Local Id 1
  Remote credits 1, Local credits 0, Advertised Credits 2
  Flags: none
  Max Data Slot 255, Max Attn Slot 255, Stop Reason 0
Remote Node data:
Node ”ENG2”, Address AA00.0400.34DC, usage 1
  Timer 189,  sequence 61,  changes 247,  flags 0x0, protocol 5.1
  Recv 60/29/186,  Xmit 52/36/21,  0 Dups, 0 ReXmit
  Groups:   0
```

Table 7-7 describes the screen output for the preceding two examples. The output is divided into three sections: TTY data, session data, and remote node data. Where information on more than one session appears, there is a group of three sections for each session, preceded by a line identifying the session.

Table 7-7 Show LAT Sessions Field Descriptions

Field	Description
TTY data	Reports a summary of the LAT-oriented terminal-line specific data.
Name	Identifies the name used for a port as a port-identification string. This is reported to remote systems, which might display it in some operating-system dependent manner. This is also the value used for targets of host-initiated connections. Currently, this value is hard-wired to be the line number of the associated terminal line.
Local usage Remote usage	Indicate the current status of the terminal. The number is reported as *current/maximum*, where *current* is the current number of sessions of a given type, and *maximum* is the maximum number of sessions allowed, or zero if there is no maximum. If a terminal is being used for outgoing sessions, the local usage will be equal to the number of current LAT sessions. If the terminal is being used for incoming sessions, local usage will be disabled and the remote count and maximum will be one.
Flags	Indicate the current state of the line, and whether there are currently any queued host-initiated connections.
Command Mode flags	Report flags that are not currently used in this software release.
Config flags	Indicate the current port state as reflected by the most recent configuration message exchange.
Flow control	Lists set flow control characters.
Groups	Report the group code list currently in use for the line.
Session data	Reports various parameters about the connection.
Name	For outbound connections, indicates the name of the remote service to which the router is connected. For inbound connections, this field is currently unused.

Field	Description
Remote/Local Id	Report the slot IDs being used to uniquely identify the session multiplexed over the underlying LAT virtual circuit.
Remote/Local/advertised credits	List the number of flow control credits that the router will be sending to the host as soon as possible. The advertised credits are the number of credits that have already been extended.
Flags	Indicate transient conditions in the LAT state machine dealing with the current connection status.
Max Data Slot	Lists the maximum number of characters that can be sent in a single data slot.
Max Attn Slot	Lists the maximum amount of data that can be sent in an attention message. Because current LAT implementations only send one-byte attention messages (attention messages are used to flush buffered output), a nonzero value means that remote data flushing can be used, a zero means that it cannot.
Stop Reason	Identifies the reason that the session was stopped, if it has been stopped but not deleted. This value is usually zero, indicating that the session has not been stopped yet. If a session persists for a long period of time with a nonzero stop reason, this generally indicates a problem in the local LAT software.
Remote node data	Reports information about the remote node. The data includes the same fields as those from the **show lat nodes** output.

show lat traffic

Use the **show lat traffic** EXEC command to display information on traffic and resource utilization statistics on all active lines of a router.

show lat traffic

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Sample Display

The following is sample output from the **show lat traffic** command:

```
sloth# show lat traffic

Local host statistics:
  0/100 circuits, 0/500 sessions, 1/500 services
  100 sessions/circuit, circuit timer 80, keep-alive timer 5
Recv:   335535 messages (2478 duplicates),  161722 slots,  1950146 bytes
        0 bad circuit messages,  3458 service messages (52 used)
Xmit:   182376 messages (2761 retransmit),  146490 slots,  36085 bytes
        1 circuit timeouts
Total:  23 circuits created,  38 sessions
```

Table 7-8 describes significant fields shown in the display.

Table 7-8 Show LAT Traffic Field Descriptions

Field	Description
Local host statistics	Information about the router
circuits	The current number and maximum support number of virtual circuits
sessions	The current and maximum number of sessions
services	The current number of known remote services, and the maximum supported
sessions/circuit	The number of sessions per virtual circuit supported by the software
circuit timer	The value of the virtual-circuit timer parameter defined by the **lat vc-timer** global configuration command
keep-alive timer	The value defined by the **lat ka-timer** global configuration command
Recv	Statistics about local node receive totals
messages	The total count of virtual circuit messages received
duplicates	The number of duplicate virtual circuit messages received
slots	The number of slots received
bytes	The actual number of data bytes received
bad circuit messages	The count of invalid messages received
service messages	The number of service announcements multicast messages received

Field	Description
used	The number of multicast messages that caused the local node information to be updated
Xmit	Various transmission totals
messages	The total number of virtual-circuit messages transmitted
etransmit	The number of virtual-circuit messages retransmitted due to the lack of an acknowledgment
slots	The number of data and control slots transmitted
bytes	The actual count of user data bytes transmitted
circuit timeouts	The count of times that a virtual circuit timed out because the remote node stopped responding (due to a node failure or communications failure)
Total	The count of virtual circuits and sessions that have existed since the router booted or rebooted

show node

Use the **show node** EXEC command to display to display information about LAT nodes.

show node [all | *node-name***] [counters | status | summary]**

Syntax Description

all	(Optional) Specifies all nodes.
node-name	(Optional) Identifies the name of the node for which status is required.
counters	(Optional) Specifies the various node counters.
status	(Optional) Specifies detailed node status. This is the default if a node name is specified.
summary	(Optional) Specifies a status summary for the node. This is the default if no node name is specified.

Command Mode

EXEC

Usage Guidelines

show node command with no arguments is the same as entering **show node all summary**.

You can enter the **show node** command with either a specific node name or the keyword **all**, but not both.

You can enter the **show node** command with only one of the keywords **counters**, **status**, or **summary**. If you enter **show node** and two of these keywords without specifying a node name, the first keyword is treated as a node name, causing an error. If you enter **show node** *node-name* and two of these keywords, the second keyword will be treated as ambiguous.

The **show node** command with no further parameters shows a one-line summary of all known nodes. The **show node** command displays three different sets of information about a node: the node counters, the node status, or a one-line summary of the node status.

Note The **show node** command with a *node-name* argument but no **counters**, **status**, or **summary** keyword defaults to **show node** *node-name* **status**.

Sample Display with No Keywords

The following is sample output from the **show node** command with no further parameters (the same as **show node all summary**):

```
pt> show node

Node Name      Status       Identification
CHAOS          Reachable
MUDDY-RIVER    Reachable
```

```
TARMAC          Reachable
WHEEL           Reachable    Welcome to VAX/VMS V5.4-2
```

Table 7-9 describes significant fields shown in the display.

Table 7-9 Show Node with no Keywords Field Descriptions

Field	Description
Node Name	Lists the names of the nodes.
Status	Indicates whether the node is reachable or not.
Identification	Lists the identification string for the node.

Sample Display with a Node Name

The following is sample output from the **show node** output that defaults to **show node chaos status**. It results in a display of the detailed status of node *CHAOS*.

```
pt> show node chaos

Node: CHAOS      Address: 00-00-0C-01-05-09
LAT Protocol: V5.1   Data Link Frame Size: 1500
Identification:
Node Groups: 0
Service Name  Status     Rating   Identification
CHAOS         Available  80
```

Table 7-10 describes significant fields shown in the display.

Table 7-10 Show Node with the Node Name Specified Field Descriptions

Field	Description
Node	Lists the node name as reported by the host computer.
Address	Identifies the MAC address of the node's Ethernet interface.
LAT protocol	Lists the version of the LAT protocol used by the node.
Data Link Frame Size	Lists the size of the largest packet that can be sent to the LAT host.
Identification	Lists the identification string for the node.
Node Groups	Lists the group-code list that is advertised by the remote node it its service announcements.
Service Name	Lists the LAT service name.
Status	Indicates whether or not the node is currently available on the network.
Rating	Indicates the rating of the service: An integer from 0 to 255, with the highest number being the preferred service. Used for load balancing.

Sample Display with the Counters Keyword

The following is sample output for the counter information for a specific node.

```
pt> show node tarmac counters

Node: tarmac
Seconds Since Zeroed: 100 Multiple Node Addresses: 0
Messages Received: 0 Duplicates Received: 0
```

```
Messages Transmitted: 0 Messages Re-transmitted: 0
Slots Received: 0 Illegal Messages Received: 0
Slots Transmitted: 0 Illegal Slots Received: 0
Bytes Received: 0 Solicitations Accepted: 0
Bytes Transmitted: 0 Solicitations Rejected: 0
```

Additional Command Examples

In the following example, the keyword **status** is treated as the node name:

```
pt> show node status counters

Local -710- Node STATUS not known
```

In the following example, the keyword **counters** is treated as ambiguous:

```
pt> show node lager status counters

Local -702- Keyword "COUNTERS" not known or ambiguous
```

show service

Use the **show service** EXEC command on a router to display specific LAT learned services.

show service [*service-name*]

Syntax Description

service-name	(Optional) The name of a specific LAT service.

Command Mode

EXEC

Usage Guidelines

The **show service** command without a service name displays a list of known LAT learned services. When entered with the *service-name* argument, it displays a more detailed status of the named service. If no LAT learned service by the specified name is known, then a lookup is done for an IP host of that name.

Sample Display

The following is sample output from the **show service** command:

```
pt> show service

Service Name  Status      Identification
BLUE          Available   Welcome to VAX/VMS V5.4
CHAOS         Available
MRL12         Available
MUDDY-RIVER   Available
STELLA-BLUE   Available   Welcome to VAX/VMS V5.4
```

The following display shows sample **show service** output for a specific service.

```
pt> show service blue

Service BLUE - Available
Node Name   Status    Rating    Identification
BLUE        reachable 84        Welcome to VAX/VMS V5.4
```

Table 7-11 describes significant fields shown in the two previous displays.

Table 7-11 Show Service Field Descriptions

Field	Description
Service	Name of the service.
Node Name	Name of the nodes advertising the service.
Status	Status of the service: Available or Unknown when command is entered without a service name. Available, Unknown, Initializing, or Unreachable when command is entered with a service name.
Rating	Rating of the service: An integer from 0 to 255, with the highest number being the preferred service. Used for load balancing.
Identification	Identification string.

CHAPTER 8

TN3270 Configuration Commands

TN3270 terminal emulation software allows any terminal to be used as an IBM 3270-type terminal. Use the commands in this chapter to configure IBM 3270 terminal emulation on your router. For configuration information and examples, refer to the chapter "Configuring TN3270" earlier in this publication. For information about establishing TN3270 connections, refer to the *Cisco Access Connection Guide*.

keymap

To define specific characteristics of keyboard mappings, use the **keymap** global configuration command. To remove the named keymap from the current image of the configuration file, use the **no** form of this command.

keymap *keymap-name keymap-entry*
no keymap *keymap-name*

Syntax Description

keymap-name	Name of the file containing the keyboard mappings. The name can be up to 32 characters long and must be unique.
keymap-entry	Commands that define the keymap.

Default

VT100 keyboard emulation

Command Mode

Global configuration

Usage Guidelines

The **keymap** command maps individual keys on a non-3270 keyboard to perform the function defined for the 3270 keyboard. Use the EXEC command **show keymap** to test for the availability of a keymap.

The guidelines for creating a keymap file follow.

The Keymap Entry Structure

A keymap consists of an entry for a keyboard. The first part of an entry lists the names of the keyboards that use that entry. These names will often be the same as in the ttycaps file; however, often the terminals from various ttycap entries will all use the same keymap entry. For example, both 925 and 925vb (for 925 with visual bells) would probably use the same keymap entry. Additionally, there are occasions when it is necessary to specify a keyboard name as the name of the entry (for example, if a user requires a custom key layout).

After the names, which are separated by vertical bars (|), comes a left brace ({), the text that forms the definitions, and a right brace (}). Each definition consists of a reserved keyword, which identifies the 3270 function, followed by an equal sign (=), followed by the various ways to generate this particular function, followed by a semicolon (;). Each alternative way to generate the function is a sequence of ASCII characters enclosed inside single quotes (“); the alternatives are separated by vertical bars (|). Inside the single quotes, a few characters are special. A caret (^) specifies that the next character is a control (Ctrl) character. The two-character string caret symbol-a (^a) represents Ctrl-a. The caret symbol-A sequence (^A) generates the same code as caret symbol-a (^a). To generate Delete (or DEL), enter the caret symbol-question mark (^?) sequence.

Note The Ctrl-caret symbol combination (Ctrl-^), used to generate a hexadecimal 1E, is represented as two caret symbols in sequence (^^)—not as a caret-backslash-caret combination (^\^).

In addition to the caret, a letter might be preceded by a backslash (\). Because this has little effect for most characters, its use is usually not recommended. In the case of a single quote ('), the backslash prevents that single quote from terminating the string. In the case of a caret (^), the backslash prevents the caret from having its special meaning. To have the backslash be part of the string, it is necessary to place two backslashes (\\) in the keymap. Table 8-1 lists other supported special characters.

Table 8-1 Special Characters Supported by TN3270 Keymap Capability

Character	Description
\E	Escape character
\n	Newline
\t	Tab
\r	Carriage return

It is not necessary for each character in a string to be enclosed within single quotes. For example, \E\E\E means three escape characters.

To enter a keymap, provide a unique name for it and explicitly define all special keys you intend to include in it within curly brackets. Also, except for the last line, each line must be terminated with a backslash symbol (\). The last line ends with a right brace (}) symbol and an end-of-line character.

Keymap Restrictions

When emulating IBM-style 3270 terminals, a mapping must be performed between sequences of keys pressed at a user's (ASCII) keyboard and the keys available on a 3270-type keyboard. For example, a 3270-type keyboard has a key labeled EEOF that erases the contents of the current field from the location of the cursor to the end. To accomplish this function, the terminal user and a program emulating a 3270-type keyboard must agree on what keys will be typed to invoke the function. The requirements for these sequences follow:

- The first character of the sequence must be outside of the standard ASCII printable characters.
- No sequence can be a complete subset of another sequence (although sequences can share partial elements).

Following are examples of acceptable keymap entries:

```
pfk1 = '\E1';
pfk2 = '\E2';
```

Following are examples of unacceptable keymap entries:

```
pfk1 = '\E1';
pfk11 = '\E11';
```

In the acceptable example, the keymap entry for *pfk1* is not completely included in the keymap entry for *pfk2*. By contrast, in the unacceptable, or conflicting keymap pair, the sequence used to represent *pfk1* is a complete subset of the sequence used to represent *pfk11*. Refer to the keymap entry provided later in this section for an example of how various keys can be represented to avoid this kind of conflict.

Table 8-2 is the list of 3270 key names that are supported in this keymap. Note that some of the keys do not really exist on a 3270-type keyboard. An unsupported function will cause the router to send a (possibly visual) bell sequence to the user's terminal.

Table 8-2 3270 Key Names Supported by Default Keymap

3270 Key Name	Functional Description
LPRT[1]	Local print
DP	Dup character
FM	Field mark character
CURSEL	Cursor select
CENTSIGN	EBCDIC cent sign
RESHOW	Redisplay the screen
EINP	Erase input
EEOF	Erase end of field
DELETE	Delete character
INSRT	Toggle insert mode
TAB	Field tab
BTAB	Field back tab
COLTAB	Column tab
COLBAK	Column back tab
INDENT	Indent one tab stop
UNDENT	Unindent one tab stop
NL	New line
HOME	Home the cursor
UP	Up cursor
DOWN	Down cursor
RIGHT	Right cursor
LEFT	Left cursor
SETTAB	Set a column tab
DELTAB	Delete a column tab
SETMRG	Set left margin
SETHOM	Set home position
CLRTAB	Clear all column tabs
APLON[1]	Apl on
APLOFF[1]	Apl off
APLEND[1]	Treat input as ASCII

3270 Key Name	Functional Description
PCON[1]	Xon/xoff on
PCOFF[1]	Xon/xoff off
DISC	Disconnect (suspend)
INIT[1]	New terminal type
ALTK[1]	Alternate keyboard dvorak
FLINP	Flush input
ERASE	Erase last character
WERASE	Erase last word
FERASE	Erase field
SYNCH	We are in synch with the user
RESET	Reset key, unlock keyboard
MASTER_RESET	Reset, unlock, and redisplay
XOFF[1]	Please hold output
XON[1]	Please give me output
WORDTAB	Tab to beginning of next word
WORDBACKTAB	Tab to beginning of current/last word
WORDEND	Tab to end of current/next word
FIELDEND	Tab to last nonblank of current/next unprotected (writable) field
PA1	Program attention 1
PA2	Program attention 2
PA3	Program attention 3
CLEAR	Local clear of the 3270 screen
TREQ	Test request
ENTER	Enter key
PFK1 to PFK30	Program function key 1 to program function key 30

1. Not supported by Cisco's TN3270 implementation.

Table 8-3 describes the proper keys used to emulate each 3270 function when using the default key mapping supplied.

Table 8-3 Keys Used to Emulate Each 3270 Function with Default Keymap

Command Keys	IBM 3270 Key	Default Keys
Cursor Movement Keys	New Line	Ctrl-n or Home
	Tab	Ctrl-i
	Back Tab	Ctrl-b
	Back Tab	Ctrl-b
	Cursor Left	Ctrl-h
	Cursor Right	Ctrl-l
	Cursor Up	Ctrl-k
	Cursor Down	Ctrl-j or LINE FEED
Edit Control Keys	Delete Char	Ctrl-d or RUB
	Erase EOF	Ctrl-e
	Erase Input	Ctrl-w
	Insert Mode	ESC Space[1]
	End Insert	ESC Space
Program Function Keys	PF1	ESC 1
	PF2	ESC 2
	...	...
	PF10	ESC 0
	PF11	ESC -
	PF12	ESC =
	PF13	ESC !
	PF14	ESC @
	...	...
	PF24	ESC +
Program Attention Keys	PA1	Ctrl-p 1
	PA2	Ctrl-p 2
	PA3	Ctrl-p 3
Local Control Keys	Reset After Error	Ctrl-r
	Purge Input Buffer	Ctrl-x
	Keyboard Unlock	Ctrl-t
	Redisplay Screen	Ctrl-v
Other Keys	Enter	Return
	Clear	Ctrl-z
	Erase current field	Ctrl-u

1. ESC refers to the Escape key.

Example

The following example is the default entry used by the TN3270 emulation software when it is unable to locate a valid keymap in the active configuration image. Table 8-2 lists the key names supported by the default Cisco TN3270 keymap.

```
ciscodefault{
clear = '^z';\
flinp = '^x';\
enter = '^m';\
delete = '^d' | '^?';\
synch = '^r';\
reshow = '^v';\
eeof = '^e';\
tab = '^i';\
btab = '^b';\
nl = '^n';\
left = '^h';\
right = '^l';\
up = '^k';\
down = '^j';\
einp = '^w';\
reset = '^t';\
ferase = '^u';\
insrt = '\E ';\
pa1 = '^p1'; pa2 = '^p2'; pa3 = '^p3';\
pfk1 = '\E1'; pfk2 = '\E2'; pfk3 = '\E3'; pfk4 = '\E4';\
pfk5 = '\E5'; pfk6 = '\E6'; pfk7 = '\E7'; pfk8 = '\E8';\
pfk9 = '\E9'; pfk10 = '\E0'; pfk11 = '\E-'; pfk12 = '\E=';\
pfk13 = '\E!'; pfk14 = '\E@'; pfk15 = '\E#'; pfk16 = '\E$';\
pfk17 = '\E%'; pfk18 = '\E'; pfk19 = '\E&'; pfk20 = '\E*';\
pfk21 = '\E('; pfk22 = '\E)'; pfk23 = '\E_'; pfk24 = '\E+';\
}
```

Related Commands

keymap-type
show keymap
terminal-type

keymap-type

To specify the keyboard map for a terminal connected to the line, use the **keymap-type** line configuration command. To reset the keyboard type for the line to the default, use the **no** form of this command.

keymap-type *keymap-name*
no keymap-type

Syntax Description

keymap-name	Name of a keymap defined within the configuration file of the router. The TN3270 terminal-type negotiations use the specified keymap type when setting up a connection with the remote host.

Default

VT100

Command Mode

Line configuration

Usage Guidelines

The **keymap-type** command must follow the corresponding **keymap** global configuration entry in the configuration file.

Setting the keyboard to a different keymap requires that a keymap be defined with the router's configuration either by obtaining a configuration file over the network that includes the keymap definition or by defining the keyboard mapping using the global configuration command **keymap**.

Use the EXEC command **show keymap** to test for the availability of a keymap.

Example

The following example sets the keyboard mapping to a keymap named *vt100map*:

```
line 3
keymap-type vt100map
```

Related Commands

keymap
show keymap
ttycap

show keymap

Use the **show keymap** EXEC command to test for the availability of a keymap after a connection on a router takes place.

show keymap [*keymap-name* | **all**]

Syntax Description

keymap-name	(Optional) Name of the keymap.
all	(Optional) Lists the names of all defined keymaps. The name of the default keymap is not listed.

Command Mode

EXEC

Usage Guidelines

The router searches for the specified keymap in its active configuration image and lists the complete entry if found. If the keymap is not found, an appropriate “not found” message appear.

If you do not use any arguments with the **show keymap** command, then the keymap currently used for the terminal is displayed.

Sample Display

The following is sample output from the **show keymap** command:

```
pt# show keymap

ciscodefault { clear = '^z'; flinp = '^x'; enter = '^m';\
      delete = '^d' | '^?';\
      synch = '^r'; reshow = '^v'; eeof = '^e'; tab = '^i';\
      btab = '^b'; nl = '^n'; left = '^h'; right = '^l';\
      up = '^k'; down = '^j'; einp = '^w'; reset = '^t';\
      xoff = '^s'; xon = '^q'; escape = '^c'; ferase = '^u';\
      insrt = '\E ';\
      pa1 = '^p1'; pa2 = '^p2'; pa3 = '^p3';\
      pfk1 = '\E1'; pfk2 = '\E2'; pfk3 = '\E3'; pfk4 = '\E4';\
      pfk5 = '\E5'; pfk6 = '\E6'; pfk7 = '\E7'; pfk8 = '\E8';\
      pfk9 = '\E9'; pfk10 = '\E0'; pfk11 = '\E-'; pfk12 = '\E=';\
      pfk13 = '\E!'; pfk14 = '\E@'; pfk15 = '\E#'; pfk16 = '\E$';\
      pfk17 = '\E%'; pfk18 = '\E\^'; pfk19 = '\E&'; pfk20 = '\E*';\
      pfk21 = '\E('; pfk22 = '\E)'; pfk23 = '\E_'; pfk24 = '\E+';\
}
```

show tn3270 ascii-hexval

To determine ASCII-hexadecimal character mappings, use the **show tn3270 ascii-hexval** EXEC command.

show tn3270 ascii-hexval

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Usage Guidelines

Use the **show tn3270 ascii-hexval** command to display the hexadecimal value of a character on your keyboard. After entering the **show tn3270 ascii-hexval** command, you are prompted to press a key. The hexadecimal value of the ASCII character is displayed. This command is useful for users who do not know the ASCII codes associated with various keys or do not have manuals for their terminals.

Sample Displays

The following are sample output from the **show tn3270 ascii-hexval** command:

```
chaff> show tn3270 ascii-hexval
Press key> 7 - hexadecimal value is 0x37.

chaff> show tn3270 ascii-hexval
Press key> f - hexadecimal value is 0x66.

tarmac> show tn3270 ascii-hexval
Press key> not printable - hexadecimal value is 0xD.
```

Related Commands

tn3270 character-map
show tn3270 character-map

show tn3270 character-map

To display ASCII to EBCDIC character mappings, use the **show tn3270 character-map** EXEC command.

show tn3270 character-map {**all** | *ebcdic-in-hex*}

Syntax Description

all	Displays all nonstandard character mappings.
ebcdic-in-hex	EBCDIC character whose ASCII mapping you want to display.

Command Mode

EXEC

Sample Displays

The following is sample output of the **show tn3270** character map command with the **all** keyword:

```
tarmac# show tn3270 character-map all

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

Related Commands

tn3270 character-map
show tn3270 ascii-hexval

show ttycap

To test for the availability of a ttycap after a connection on a router takes place, use the **show ttycap** EXEC command.

show ttycap [*ttycap-name* | **all**]

Syntax Description

ttycap-name	(Optional) Name of a ttycap.
all	(Optional) Lists the names of all defined ttycaps. The name of the default keymap is not listed.

Command Mode

EXEC

Usage Guidelines

The router searches for the specified ttycap in its active configuration image, and lists the complete entry if found. If it is not found, an appropriate “not found” message appear.

If you do not include any arguments with the **show ttycap** command, then the current keymap used for the terminal is displayed.

Sample Displays

The following sample output from the **show ttycap** command displays current output for the current keymap used by the terminal:

```
pt# show ttycap

d0|vt100|vt100-am|vt100am|dec vt100:do=^J:co#80:li#24:\
cl=50^[[;H^[[2J:bs:am:cm=5^[[%i%d;%dH:nd=2^[[C:up=2^[[A:\
ce=3^[[K:so=2^[[7m:se=2^[[m:us=2^[[4m:ue=2^[[m:md=2^[[1m:\
me=2^[[m:ho=^[[H:xn:sc=^[7:rc=^[8:cs=^[[%i%d;%dr:
```

The following sample output from the **show ttycap all** command displays current output for all the keymaps:

```
pt# show ttycap all

ttycap3    d0|vt100|vt100-am|vt100am|dec vt100
ttycap2    dl|vt200|vt220|vt200-js|vt220-js|dec vt200 series with jump scroll
ttycap1    ku|h19-u|h19u|heathkit with underscore cursor
```

The following sample output from the **show ttycap ttycap1** command displays current output for the ttycap1:

```
pt# show ttycap ttycap1

ttycap1  ku|h19-u|h19u|heathkit with underscore cursor:\:vs@:ve@:tc=h19-b:\
        :al=1*\EL:am:le=^H:bs:cd=\EJ:ce=\EK:cl=\EE:cm=\EY%+ %+\
        :co#80:dc=\EN:\:dl=1*\EM:do=\EB:ei=\EO:ho=\EH\
        :im=\E@:li#24:mi:nd=\EC:as=\EF:ae=\EG:\
        :ms:pt:sr=\EI:se=\Eq:so=\Ep:up=\EA:vs=\Ex4:ve=\Ey4:\
        :kb=^h:ku=\EA:kd=\EB:kl=\ED:kr=\EC:kh=\EH:kn#8:ke=\E>:ks=\E=:\
```

```
:k1=\ES:k2=\ET:k3=\EU:k4=\EV:k5=\EW:\
:l6=blue:l7=red:l8=white:k6=\EP:k7=\EQ:k8=\ER:\
:es:hs:ts=\Ej\Ex5\Ex1\EY8%+ \Eo:fs=\Ek\Ey5:ds=\Ey1:
```

terminal-type

To specify the type of terminal connected to the line, use the **terminal-type** line configuration command. To reset the terminal type for the line to the default, use the **no** form of this command.

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

Syntax Description

terminal-name	Name of a termcap defined within the configuration file.

Default

VT100

Command Mode

Line configuration

Usage Guidelines

The **terminal-type** command must follow the corresponding **ttycap** global configuration entry in the configuration file. Use the EXEC command **show ttycap** to test for the availability of a ttycap.

The TN3270 terminal-type negotiations use the specified terminal type when setting up a connection with the remote host.

Setting the terminal type to VT220 requires that the ttycap be defined within the router's configuration either by obtaining a configuration file over the network that includes the ttycap definition, or by defining the ttycap mapping via the global configuration command **ttycap**.

Example

The following example sets the terminal line 5 to type VT220:

```
line 5
terminal-type VT220
```

Related Commands

keymap
show ttycap
ttycap

tn3270 8bit display

To configure the communication server to use the mask set by the **data-character-bits** {**7** | **8**} line configuration command or the **terminal data-character bits** {**7** | **8**} EXEC command, use the **tn3270 8bit display** line configuration command. To restore the default 7-bit mask used for TN3270 connections, use the **no** form of this command.

tn3270 8bit display
no tn3270 8bit display

Syntax Description

This command has no arguments or keywords.

Default

Disabled

Command Mode

Line configuration

Usage Guidelines

Use the **tn3270-character-map** command to map between extended EBCDIC or extended ASCII characters.

Example

The following example configures the communication server to use the mask set by the **data-character-bits** line configuration and EXEC commands on line 5:

```
line 5
tn3270 8bit display
```

Related Commands

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication. Two daggers (††) indicate that the command is documented in the *Cisco Access Connection Guide*.

data-character-bits†
terminal data-character-bits††

tn3270 8bit transparent-mode

To configure the router to use an 8-bit mask, use the **tn3270 8bit transparent-mode** command. The **no** form of this command reverts to the default 7-bit mask used for TN3270 connections.

tn3270 8bit transparent-mode
no tn3270 8bit transparent-mode

Syntax Description

This command has no arguments or keywords.

Default

Disabled

Command Mode

Line configuration

Usage Guidelines

This command is needed if you are using a file transfer protocol such as Kermit in 8-bit mode or you are using 8-bit graphics, both of which rely on transparent mode.

Example

The following example configures the router to use the mask set by the **data-character-bits** line configuration and EXEC commands on line 5:

```
line 5
tn3270 8bit transparent-mode
```

Related Commands

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication. Two daggers (††) indicate that the command is documented in the *Cisco Access Connection Guide*.

data-character-bits †
terminal data-character-bits ††

tn3270 character-map

To create a two-way binding between EBCDIC and ASCII characters, use the **tn3270 character-map** global configuration command. To restore default character mappings, use the **no** form of this command.

tn3270 character-map *ebcdic-in-hex ascii-in-hex*
no tn3270 character-map {**all** | *ebcdic-in-hex*} [*ascii-in-hex*]

Syntax Description

ebcdic-in-hex	Hexadecimal value of an EBCDIC character.
ascii-in-hex	Hexadecimal value of an ASCII character.
all	Indicates all character mappings.

Default

Disabled

Command Mode

Global configuration

Usage Guidelines

Use this command to print EBCDIC international characters that are not normally printed, including umlauts (¨) and tildes (~). The command first restores default mapping for both EBCDIC and ASCII characters. In the **no** form of the command, the **all** keyword resets all character mappings to Cisco defaults.

Table 8-4 shows the default ASCII and EBCDIC character mappings, in decimal and hexadecimal format.

Table 8-4 Default ASCII, EBCDIC Character Mappings

Character	ASCII Decimal	ASCII Hexadecimal	EBCDIC Decimal	EBCDIC Hexadecimal
!	33	0x21	90	0x5a
“	34	0x22	127	0x7f
#	35	0x23	123	0x7b
$	36	0x24	91	0x5b
%	37	0x25	108	0x6c
&	38	0x26	80	0x50
’	39	0x27	125	0x7d
(	40	0x28	77	0x4d
)	41	0x29	93	0x5d
*	42	0x2a	92	0x5c

Character	ASCII Decimal	ASCII Hexadecimal	EBCDIC Decimal	EBCDIC Hexadecimal
+	43	0x2b	78	0x4e
,	44	0x2c	107	0x6b
-	45	0x2d	96	0x60
.	46	0x2e	75	0x4b
/	47	0x2f	97	0x61
0	48	0x30	240	0xf0
1	49	0x31	241	0xf1
2	50	0x32	242	0xf2
3	51	0x33	243	0xf3
4	52	0x34	244	0xf4
5	53	0x35	245	0xf5
6	54	0x36	246	0xf6
7	55	0x37	247	0xf7
8	56	0x38	248	0xf8
9	57	0x39	249	0xf9
:	58	0x3a	122	0x7a
;	59	0x3b	94	0x5e
<	60	0x3c	76	0x4c
=	61	0x3d	126	0x7e
>	62	0x3e	110	0x6e
?	63	0x3f	111	0x6f
@	64	0x40	124	0x7c
A	65	0x41	193	0xc1
B	66	0x42	194	0xc2
C	67	0x43	195	0xc3
D	68	0x44	196	0xc4
E	69	0x45	197	0xc5
F	70	0x46	198	0xc6
G	71	0x47	199	0xc7
H	72	0x48	200	0xc8
I	73	0x49	201	0xc9
J	74	0x4a	209	0xd1
K	75	0x4b	210	0xd2
L	76	0x4c	211	0xd3
M	77	0x4d	212	0xd4
N	78	0x4e	213	0xd5
O	79	0x4f	214	0xd6
P	80	0x50	215	0xd7

Character	ASCII Decimal	ASCII Hexadecimal	EBCDIC Decimal	EBCDIC Hexadecimal
Q	81	0x51	216	0xd8
R	82	0x52	217	0xd9
S	83	0x53	226	0xe2
T	84	0x54	227	0xe3
U	85	0x55	228	0xe4
V	86	0x56	229	0xe5
W	87	0x57	230	0xe6
X	88	0x58	231	0xe7
Y	89	0x59	232	0xe8
Z	90	0x5a	233	0xe9
[	91	0x5b	173	0xad
\	92	0x5c	224	0xe0
]	93	0x5d	189	0xbd
^	94	0x5e	95	0x5f
_	95	0x5f	109	0x6d
`	96	0x60	121	0x79
a	97	0x61	129	0x81
b	98	0x62	130	0x82
c	99	0x63	131	0x83
d	100	0x64	132	0x84
e	101	0x65	133	0x85
f	102	0x66	134	0x86
g	103	0x67	135	0x87
h	104	0x68	136	0x88
i	105	0x69	137	0x89
j	106	0x6a	145	0x91
k	107	0x6b	146	0x92
l	108	0x6c	147	0x93
m	109	0x6d	148	0x94
n	110	0x6e	149	0x95
o	111	0x6f	150	0x96
p	112	0x70	151	0x97
q	113	0x71	152	0x98
r	114	0x72	153	0x99
s	115	0x73	162	0xa2
t	116	0x74	163	0xa3
u	117	0x75	164	0xa4
v	118	0x76	165	0xa5

Character	ASCII Decimal	ASCII Hexadecimal	EBCDIC Decimal	EBCDIC Hexadecimal
w	119	0x77	166	0xa6
x	120	0x78	167	0xa7
y	121	0x79	168	0xa8
z	122	0x7a	169	0xa9
{	123	0x7b	192	0xc0
|	124	0x7c	79	0x4f
}	125	0x7d	208	0xd0
~	126	0x7e	161	0xa1

Example

The following example creates a two-way binding between an EBCDIC character and an ASCII character:

```
tn3270 character-map 0x81 0x78
```

Related Commands

show tn3270 character-map
show tn3270 ascii-hexval

tn3270 datastream

To permit the use of extended features, use the **tn3270 datastream** line configuration command. Use the **no** form of this command to restore the default setting of support for normal features.

tn3270 datastream {**extended** | **normal**}

Syntax Description

extended	Appends "-E" to the terminal type string sent to the IBM host.
normal	Restores the default for support of normal features.

Default

Normal

Command Mode

Line configuration

Usage Guidelines

Use the **tn3270 datastream** command to allow the use of extended features over routers that support such features.

Example

The following example configures the router to add "-E" to the terminal type string sent to the IBM host:

```
line 5
tn3270 datastream extended
```

tn3270 null-processing

To specify the processing of null characters on a terminal screen, use the **tn3270 null-processing** line configuration command. Use the **no** form of this command to restore the default of improved null-character processing.

tn3270 null-processing {**3270** | **improved**}

Syntax Description

3270	Emulating a 3278x terminal, specifies the compression of null characters out of strings.
improved	Emulating a 7171 controller, specifies the conversion of null characters into spaces.

Default

Improved

Command Mode

Line configuration

Usage Guidelines

If a user enters data at a terminal keyboard, then uses an arrow key to move the cursor across the display screen and enters more data, the blanks spaces are filled with nulls.

Use this command to manage the processing of null characters on the terminal screen. If you use the 3270 parameter, all null characters are compressed out of the character string. The improved parameter converts all null characters to spaces.

Example

The following example configures the router to convert null characters to spaces:

```
line 5
tn3270 null-processing improved
```

tn3270 reset-required

To specify the locking of the user's keyboard until the Reset key is pressed, use the **tn3270 reset-required** line configuration command. Use the **no** form of this command to restore the default setting so that pressing the Reset key is not required.

tn3270 reset-required
no tn3270 reset-required

Syntax Description

This command has no arguments or keywords.

Default

Disabled

Command Mode

Line configuration

Usage Guidelines

Use this command to emulate the behavior of the 3278x terminal, on which the keyboard is locked and further input is denied until the Reset key is pressed. The majority of 3270 terminal emulations leave the keyboard unlocked and remove any error message at the next keystroke.

Example

The following example configures the router to lock the user's keyboard until the Reset key is press:

```
line 5
tn3270 reset-required
```

ttycap

To define characteristics of a terminal emulation file, use the **ttycap** global configuration command. To delete any named ttycap entry from the configuration file, use the **no** form of this command.

ttycap *ttycap-name termcap-entry*
no ttycap *ttycap-name*

Syntax Description

termcap-name	Name of a file. It can be up to 32 characters long and must be unique.
termcap-entry	Commands that define the ttycap. Consists of two parts (see Usage Guidelines for details).

Default

VT100 terminal emulation

Command Mode

Global configuration

Usage Guidelines

Use the EXEC command **show ttycap** to test for the availability of a ttycap.

Note Do not name a ttycap entry filename "default" or the router will adopt the newly defined entry as the default.

The *termcap-entry* consists of two parts: a *name* portion and a *capabilities* portion.

The *name* portion is a series of names that can be used to refer to a specific terminal type. Generally, these names should represent commonly recognized terminal names (such as VT100 and VT200). Multiple names can be used. Each name is separated by a vertical bar symbol (|). The series is terminated by a colon symbol (:).

The following example illustrates a name specification for a VT100 termcap.

```
d0|vt100|vt100-am|vt100am|dec vt100:
```

The *capabilities* portion of the termcap-entry consists of a sequence of termcap capabilities. These capabilities can include boolean flags, string sequences, or numeric sequences. Each individual capability is terminated using a colon symbol (:).

A *Boolean flag* can be set to true by including the two-character capability name in the termcap entry. The absence of any supported flag results in the flag being set to false.

The following is an example of a backspace Boolean flag:

```
bs:
```

A *string sequence* is a two-character capability name followed by an equal sign (=) and the character sequence.

The following example illustrates the capability for homing the cursor:

```
ho=\E[H:
```

The sequence \E represents the ESC character.

Control characters can be represented in *string sequences* by entering a two-character sequence starting with a caret symbol (^), followed by the character to be used as a control character.

The following example illustrates the definition of a control character:

```
bc=^h:
```

In this example, the backspace is entered into the *termcap-entry* as the string sequence as the characters "^h."

A *numeric sequence* is a two-character capability name followed by an number symbol (#) and the number.

The following example represents the number of columns on a screen:

```
co#80:
```

Use the backslash symbol (\ ) to extend the definition to multiple lines. The end of the ttycap *termcap-entry* is specified by a colon terminating a line followed by an end-of-line character and no backslash.

For the definitions of supported Boolean-flag ttycap capabilities, see Table 8-5. For the definitions of supported string-sequence ttycap capabilities, see Table 8-6. For the definitions of supported number-sequence ttycap capabilities, see Table 8-7.

Table 8-5 Definitions of Ttycap Capabilities: Boolean Flags

Boolean Flag	Description
am	Automatic margin
bs	Terminal can backspace with bs
ms	Safe to move in standout modes
nc	No currently working carriage return
xn	NEWLINE ignored after 80 cols (Concept)
xs	Standout not erased by overwriting (Hewlett-Packard)

Table 8-6 Definitions of Ttycap Capabilities: String Sequences

String Sequence	Description
AL	Add line below with cursor sequence
bc	Backspace if not ^h
bt	Backtab sequence
ce	Clear to end of line
cl	Clear screen, cursor to upper left
cm	Move cursor to row # and col #

String Sequence	Description
cr	Carriage return sequence
cs	Change scrolling region
DL	Delete the line the cursor is on
ei	End insert mode
ho	Home, move cursor to upper left
ic	Character insert
im	Begin insert mode
is	Initialization string (typically tab stop initialization)
ll	Move cursor to lower left corner
md	Turn on bold (extra bright) character attribute
me	Turn off all character attributes
nd	Nondestructive space
nl	Newline sequence
pc	Pad character if not NULL
rc	Restore cursor position
rs	Resets terminal to known starting state
sc	Save cursor position
se	End standout mode (highlight)
so	Start standout mode (highlight)
ta	Tab
te	End programs that use cursor motion
ti	Initialization for programs that use cursor motion
uc	Underline character at cursor
ue	End underline mode
up	Move cursor up
us	Begin underline mode
vb	Visual bell
vs	Visual cursor
ve	Normal cursor

Table 8-7 Definitions of Ttycap Capabilities: Number Sequences

Number Sequence	Description
li	Lines on the screen.
co	Columns on the screen.
sg	Standout glitch, number of spaces printed when entering or leaving standout display mode.
ug	Underline glitch, number of spaces printed when entering or leaving underline mode.

Example

The following is an example of a ttycap file. See the chapter “Configuring TN3270” earlier in this publication and the *tn3270.examples* file in the Cisco *ftp.cisco.com* directory for more examples.

```
ttycap ttycap1\
d0|vt100|vt100-am|vt100am|dec vt100:do=^J:co#80:li#24:\
cl=50^[[;H^[[2J:bs:am:cm=5^[[%i%d;%dH:nd=2^[[C:up=2^[[A:\
ce=3^[[K:so=2^[[7m:se=2^[[m:us=2^[[4m:ue=2^[[m:md=2^[[1m:\
me=2^[[m:ho=^[[H:xn:sc=^[7:rc=^[8:cs=^[[%i%d;%dr:
```

Related Commands

terminal-type
keymap-type

CHAPTER 9

SLIP and PPP Configuration Commands

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

Using SLIP or PPP encapsulation over asynchronous lines is an inexpensive way of connecting PCs to a network. SLIP and PPP over asynchronous dial-up modems allow a home computer to be connected to a network without the cost of a leased line. Dial-up SLIP and PPP links can also be used for remote sites that need only occasional telecommuting or backup connectivity. Both public-domain and vendor-supported SLIP and PPP implementations are available for a variety of computer applications.

Use the commands in this chapter to configure SLIP and PPP on your router. For configuration information and examples, refer to the chapter "Configuring SLIP and PPP" earlier in this publication.

See the *Cisco Access Connection Guide* for information about SLIP and PPP user-level EXEC connection commands.

async default ip address

To set the address used on the remote (PC) side, use the **async default ip address** interface configuration command. To remove the default address from your configuration, use the **no** form of this command.

async default ip address *address*
no async default ip address

Syntax Description

address Address of the client interface.

Default

No default address is specified.

Command Mode

Interface configuration

Example

The following example specifies address 182.32.7.51 on asynchronous interface 6:

```
line 20
speed 19200
interface async 6
async default ip address 182.32.7.51
```

Related Command

async dynamic address

async dynamic address

To specify dynamic asynchronous addressing, use the **async dynamic address** interface configuration command. To disable dynamic addressing, use the **no** form of this command.

async dynamic address
no async dynamic address

Syntax Description

This command has no arguments or keywords.

Default

Dynamic addressing is disabled.

Command Mode

Interface configuration

Usage Guidelines

You can control whether addressing is dynamic (you specific the address at the EXEC level when making the connection), or whether default addressing is used (the system forces the address). If you specify dynamic addressing, the router must be in interactive mode.

It is common to configure an asynchronous interface to have a default address and to allow dynamic addressing. With this configuration, the choice between the default address or dynamic addressing is made when you enter the **slip** or **ppp** EXEC command. If you enter an address, it is used, and if you enter the **default** keyword, the default address is used.

Example

The following example shows dynamic addressing assigned to asynchronous interface 6:

```
interface ethernet 0
ip address 1.0.0.1 255.0.0.0
interface async 6
async dynamic address
```

Related Command

async default ip address

async dynamic routing

To allow the use of routing protocols on an interface, use the **async dynamic routing** interface configuration command. To disable the use of routing protocols, use the **no** form of this command.

async dynamic routing
no async dynamic routing

Syntax Description

This command has no arguments or keywords.

Default

Dynamic routing is disabled.

Command Mode

Interface configuration

Usage Guidelines

The use of routing protocols is further controlled by the use of the **/routing** keyword in the **slip** and **ppp** EXEC command.

Example

The following example shows how to enable asynchronous routing on asynchronous interface 6. The **ip tcp header-compression passive** command enables Van Jacobson TCP header compression and prevents transmission of compressed packets until a compressed packet arrives from the asynchronous link.

```
interface async 6
async dynamic routing
async dynamic address
async default ip address 1.1.1.2
ip tcp header-compression passive
ip unnumbered ethernet 0
```

Related Commands

async dynamic address
ip tcp header-compression

async mode dedicated

To place a line into dedicated asynchronous mode using SLIP or PPP encapsulation, use the **async mode dedicated** interface configuration command. To return the line to interactive mode, use the **no** form of this command.

async mode dedicated
no async mode

Syntax Description

This command has no arguments or keywords.

Default

Asynchronous mode is disabled.

Command Mode

Interface configuration

Usage Guidelines

With dedicated asynchronous network mode, the interface uses either SLIP or PPP encapsulation, depending on which **encapsulation** method is configured for the interface. An EXEC prompt does not appear, and the router is not available for normal interactive use.

If you configure a line for dedicated mode, you will not be able to use the **async dynamic address** command, because there is no user prompt.

Example

The following example assigns an IP address to an asynchronous line and places the line into network mode. Setting the stop bits to 1 enhances performance.

```
interface async 4
async default ip address 182.32.7.51
async mode dedicated
encapsulation slip

line 20
location Joe's computer
stopbits 1
speed 19200
```

Related Command

async mode interactive

async mode interactive

To return a line that has been placed into dedicated asynchronous network mode to interactive mode, thereby enabling the **slip** and **ppp** EXEC commands, use the **async mode interactive** interface configuration command. To prevent users from implementing SLIP and PPP at the EXEC level, use the **no** form of this command.

async mode interactive
no async mode

Syntax Description

This command has no arguments or keywords.

Default

Asynchronous mode is disabled.

Command Mode

Interface configuration

Usage Guidelines

Interactive mode enables the **slip** and **ppp** EXEC commands. In dedicated mode, there is no user EXEC level. You do not enter any commands, and a connection is automatically established when you log in, according to the configuration.

Example

The following example places asynchronous interface 6 into interactive asynchronous mode:

```
interface async 6
async default ip address 182.32.7.51
async mode interactive
ip unnumbered ethernet 0
```

Related Command

async mode dedicated

async-bootp

To support the extended BOOTP request specified in RFC 1084, and to specify information that will be sent in response to BOOTP requests, use the **async-bootp** global configuration command. To clear the list, use the **no** form of this command.

async-bootp *tag* [**:***hostname*] *data*
no async-bootp *tag* [**:***hostname*] *data*

Syntax Description

tag	Item being requested; expressed as a filename, an integer, or an IP dotted decimal address. See Table 9-1 in the "Usage Guidelines" section for possible values.
:*hostname*	(Optional) This entry applies only to the specified host. The argument can be either an IP address or a logical host name.
data	List of IP addresses entered in dotted decimal notation or as logical host names, a number, or a quoted string.

Default

If no extended BOOTP commands are entered, the software generates a gateway and subnet mask appropriate for the local network.

Command Mode

Global configuration

Usage Guidelines

Each of the *tag* keyword-argument pairs is a field that can be filled in and sent in response to BOOTP requests from clients.

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

Use the **show async bootp** EXEC command to list the configured parameters. BOOTP works for both SLIP and PPP.

Table 9-1 Supported Extended BOOTP Requests

Keyword and Argument Pair	Use
bootfile	Server boot file from which to download the boot program. Use the optional **:***hostname* and *data* arguments to specify the host or hosts.
subnet-mask *mask*	Dotted decimal address specifying the network and local subnetwork mask (as defined by RFC 950).
time-offset *offset*	A signed 32-bit integer specifying the time offset of the local subnetwork in seconds from Coordinated Universal Time.
gateway *address*	Dotted decimal address specifying the IP addresses of gateways for this subnetwork. A preferred gateway should be listed first.
time-server *address*	Dotted decimal address specifying the IP address of time servers (as defined by RFC 868).
ien116-server *address*	Dotted decimal address specifying the IP address of name servers (as defined by IEN 116).
dns-server *address*	Dotted decimal address specifying the IP address of Domain Name Servers (as defined by RFC 1034).
log-server *address*	Dotted decimal address specifying the IP address of an MIT-LCS UDP log server.
quote-server *address*	Dotted decimal address specifying the IP address of Quote of the Day servers (as defined in RFC 865).
lpr-server *address*	Dotted decimal address specifying the IP address of Berkeley UNIX Version 4 BSD servers.
impress-server *address*	Dotted decimal address specifying the IP address of Impress network image servers.
rlp-server *address*	Dotted decimal address specifying the IP address of Resource Location Protocol (RLP) servers (as defined in RFC 887).
hostname *name*	Name of the client (which might or might not be domain qualified, depending upon the site).
bootfile-size *value*	Two-octet value specifying the number of 512 octet (byte) blocks in the default boot file.

Examples

The following example specifies different boot files: one for a PC and one for a Macintosh. With this configuration, a BOOTP request from the host on 128.128.1.1 results in a reply listing the boot filename as *pcboot.* A BOOTP request from the host named *mac* results in a reply listing the boot filename as *macboot.*

```
async-bootp bootfile :128.128.1.1 “pcboot”
async-bootp bootfile :mac “macboot”
```

The following example specifies a subnet mask of 255.255.0.0:

```
async-bootp subnet-mask 255.255.0.0
```

The following example specifies a negative time offset of the local subnetwork of -3600 seconds:

```
async-bootp time-offset -3600
```

The following example specifies the IP address of a time server:

```
async-bootp time-server 128.128.1.1
```

Related Command

show async bootp

clear line

To return a line to its idle state, enter the **clear line** privileged EXEC command at the system prompt.

clear line *line-number*

Syntax Description

line-number Asynchronous line port number assigned with the **interface async** command.

Command Mode

Privileged EXEC

Usage Guidelines

Normally, this command returns the line to its conventional function as a terminal line, with the interface left in a "down" state.

Example

The following example shows how to use the **clear line** command to return serial interface 5 to its idle state:

```
clear line 5
```

encapsulation

To configure SLIP or PPP encapsulation as the default on an asynchronous interface, use the **encapsulation** interface configuration command. To disable encapsulation, use the **no** form of this command.

encapsulation {**slip** | **ppp**}
no encapsulation {**slip** | **ppp**}

Syntax Description

slip	Specifies SLIP encapsulation for an interface configured for dedicated asynchronous mode or DDR.
ppp	Specifies PPP encapsulation for an interface configured for dedicated asynchronous mode or DDR.

Default

SLIP encapsulation is enabled by default.

Command Mode

Interface configuration

Usage Guidelines

On lines configured for interactive use, you select encapsulation when you establish a connection with the **slip** or **ppp** EXEC command.

IP Control Protocol (IPCP) is the part of PPP that brings up and configures IP links. After devices at both ends of a connection communicate and bring up PPP, they bring up the control protocol for each network protocol they intend to run over the PPP link such as IP or IPX. If you have problems passing IP packets and the **show interface** command shows that line is up, use the **debug ppp negotiations** debugging command to see if and where the negotiations are failing. You might have different versions of software running, or different versions of PPP, in which case you might need to upgrade your software or turn off PPP option negotiations. All IPCP options as listed in RFC 1332 are supported on asynchronous lines. Only Option 2, TCP/IP header compression, is supported on synchronous interfaces.

PPP echo requests are used as keepalives to detect line failure. The **no keepalive** command can be used to disable echo requests. For more information about the **no keepalive** command, refer to the chapter "IP Routing Protocols Commands" in the *Router Products Command Reference*, and the chapter "Configuring IP Routing Protocols" in the *Router Products Configuration Guide*.

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

Note Disable software flow control on SLIP and PPP lines.

Example

In the following example, asynchronous interface 1 is configured for PPP encapsulation:

```
config
interface async 1
encapsulation ppp
```

Related Commands

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

keepalive†

hold-queue

To limit the size of the IP output queue, use the **hold-queue** interface configuration command. To return the output queue to the default size, use the **no** form of this command.

hold-queue *packets*
no hold-queue

Syntax Description

packets	Maximum number of packets. The range of values is 0 through 65535.

Default

10 packets (default for asynchronous interfaces only)

Command Mode

Interface configuration

Usage Guidelines

The default of 10 packets allows the router to queue a number of back-to-back routing updates. This is the default for asynchronous interfaces only; other media types have different defaults.

The hold queue stores packets received from the network that are waiting to be sent to the client. It is recommended that the queue size not exceed ten packets on asynchronous interfaces. For most other interfaces, the queue length should not exceed 100 packets.

Example

The following example changes the packet queue length of a line to five packets:

```
interface async 2
async default ip address 182.32.7.5
hold-queue 5
```

interface

To specify the interface you want to configure, use the **interface** global configuration command. To clear the interface configuration, use the **no** form of this command.

interface *type number*
no interface

Syntax Description

type Interface type.

number Interface number. See Table 9-2 for a list of interface numbers by router model.

Default

No interface is specified by default; you must specify an interface to configure it.

Command Mode

Global configuration

Usage Guidelines

Table 9-2 Lists Interface Numbers by Router Model

Router Model	Interface Number
508-CS	1 to 8
516-CS	1 to 16
ASM-CS (fully configured)	1 to 113
2509 or 2510	1 to 8
2511 or 2512	1 to 16

Example

The following example specifies asynchronous interface 1:

```
interface async 1
```

ip access-group

To configure an access list to be used for packets transmitted to and from the asynchronous host, use the **ip access-group** interface configuration command. To disable control over packets transmitted to or from an asynchronous host, use the **no ip access-group** command.

ip access-group *access-list-number* {**in** | **out**}
no ip access-group *access-list-number*

Syntax Description

access-list-number	Assigned IP access list number.
in	Defines access control on packets transmitted from the asynchronous host.
out	Defines access control on packets being sent to the asynchronous host.

Default

Disabled

Command Mode

Interface configuration

Usage Guidelines

With this command enabled, the IP destination address of each packet is run through the access list for acceptability and is either dropped or passed.

Example

The following example assumes that users are restricted to certain servers designated as SLIP or PPP servers, but that normal terminal users can access anything on the local network:

```
! access list for normal connections
access-list 1 permit 131.108.0.0 0.0.255.255
!
! access list for SLIP packets.
access-list 2 permit 131.108.42.55
access-list 2 permit 131.108.111.1
access-list 2 permit 131.108.55.99
!
! Specify the access list
interface async 6
async dynamic address
ip access-group 1 out
ip access-group 2 in
```

ip address

To set IP addresses for an interface, use the **ip address** interface configuration command. To remove the specified addresses, use the **no ip address** interface configuration command.

ip address *address mask* [**secondary**]
no ip address *address mask* [**secondary**]

Syntax Description

address	IP address.
mask	Network mask for the associated IP network.
secondary	(Optional) Specifies additional IP addresses.

Default

No IP addresses are specified.

Command Mode

Interface configuration

Usage Guidelines

The subnet mask must be the same for all interfaces connected to subnets of the same network. Hosts can determine subnet masks using the Internet Control Message Protocol (ICMP) *Mask Request* message. Routers respond to this request with an ICMP *Mask Reply* message.

You can disable IP processing on a particular interface by removing its IP address with the **no ip address** interface configuration command. If the router detects another host using one of its IP addresses, it will print an error message on the console.

Example

In the following example, 131.108.1.27 is the primary address and 192.31.7.17 and 192.31.8.17 are secondary addresses for asynchronous interface 1:

```
interface async 1
ip address 131.108.1.27 255.255.255.0
ip address 192.31.7.17 255.255.255.0 secondary
ip address 192.31.8.17 255.255.255.0 secondary
```

ip mtu

To specify the size of the largest IP packet, use the **ip mtu** interface configuration command. To return to the default MTU size of 1500 bytes, use the **no** form of this command.

ip mtu *bytes*
no ip mtu

Syntax Description

bytes	Maximum number of bytes. The range of values is 64 to 1000000.

Default

1500 bytes

Command Mode

Interface configuration

Example

The following example sets the packet MTU size to 200 bytes:

```
interface async 5
async default ip address 182.32.7.5
ip mtu 200
```

ip tcp header-compression

To configure Van Jacobson TCP header compression on the asynchronous link, use the **ip tcp header-compression** line configuration command. To disable header compression, use the **no** form of this command.

ip tcp header-compression [**on** | **off** | **passive**]
no ip tcp header-compression

Syntax Description

on (Optional) Turns header compression on.

off (Optional) Turns header compression off.

passive (Optional) On SLIP lines, prevents transmission of compressed packets until a compressed packet arrives from the asynchronous link, unless a user specifies SLIP on the command line. For PPP, this option functions the same as the **on** option.

Default

Header compression is on.

Command Mode

Line configuration

Usage Guidelines

Header compression data areas are initialized to handle up to 16 simultaneous TCP connections. Currently, you cannot change this number. You can only turn header compression on or off or use the **passive** keyword.

On lines configured for PPP encapsulation, the keywords **passive** and **on** cause the same behavior because, before attempting header compression, PPP automatically negotiates whether it is available at each end of the connection.

There are two ways to implement header compression when the line is configured for **ip tcp header-compression passive**:

- Enter the **/compressed** option with the **slip** EXEC commands to force the line into compressed mode. This overrides the passive setting and causes the interface to behave as if header compression is enabled.
- Enter **slip** or **slip default** and the connecting system sends compressed packets to the server. The server detects the use of compression by the connecting system and automatically enters compressed mode.

If a line is configured for passive header compression and you use the **slip** or **ppp** EXEC command to enter asynchronous mode, you will see that the interface is set to match compression status used by the host at the other end of the asynchronous line.

```
Server> slip 1.0.0.1
Password:
Entering SLIP mode.
Interface IP address is 1.0.0.1, MTU is 1500 bytes
Header compression will match your system.
```

The message “Header compression will match your system” indicates that the interface is set to match the compression status used by the host at the other end of the asynchronous line. If the line was configured to have header compression on, this line would read “Header compression is On.” Refer to the *Cisco Access Connection Guide* for more information about making SLIP and PPP connections.

Example

The following example illustrates how to enable Van Jacobson TCP header compression. The **passive** keyword prevents transmission of compressed packets until a compressed packet arrives from the IP link. Notice that asynchronous routing and dynamic addressing are also enabled.

```
interface async 6
async dynamic routing
async dynamic address
ip tcp header-compression passive
```

Related Commands

Two daggers (††) indicate that the command is documented in the *Cisco Access Connection Guide.*

ppp ††
slip ††

ip unnumbered

To conserve network resources, use the **ip unnumbered** line configuration command. To disable unnumbered interfaces, use the **no** form of this command.

ip unnumbered *type number*
no ip unnumbered

Syntax Description

type	Interface type.
number	Interface number.

Default

Disabled

Command Mode

Line configuration

Usage Guidelines

You must use either the **ip address** or **ip unnumbered** command to provide the local address for an interface.

Unnumbered interfaces do not have an address. Network resources are conserved because fewer network numbers are used and routing tables are smaller.

Whenever the unnumbered interface generates a packet (for example, a routing update), it uses the address of the specified interface as the source address of the IP packet. It also uses the address of the specified interface to determine which routing processes are sending updates over the unnumbered interface. Restrictions include the following:

- You cannot use the **ping** command to determine whether the interface is up, because the interface has no address. SNMP can be used to remotely monitor interface status.
- You cannot boot from a network (TFTP) server a bootable image over an unnumbered serial interface.
- The arguments *type* and *number* must be another interface in the network server that has an IP address, not another unnumbered interface.

Example

The following example shows how to configure asynchronous interface 6 as unnumbered:

```
interface async 6
ip unnumbered ethernet 0
```

Related Command

ip address

show async bootp

To display the parameters that have been configured for extended BOOTP requests, use the **show async bootp** privileged EXEC command.

show async bootp

Syntax Description

This command has no arguments or keywords.

Command Mode

Privileged EXEC

Sample Display

The following is sample output from the **show async bootp** command:

```
router# show async bootp

The following extended data will be sent in BOOTP responses:

bootfile (for address 128.128.1.1) "pcboot"
bootfile (for address 131.108.1.111) "dirtboot"
subnet-mask 255.255.0.0
time-offset -3600
time-server 128.128.1.1
```

Table 9-3 describes significant fields shown in the display.

Table 9-3 Show Async BOOTP Field Descriptions

Field	Description
bootfile... “pcboot”	Indicates that the boot file for address 128.128.1.1 is named pcboot.
subnet-mask 255.255.0.0	Specifies the subnet mask.
time-offset -3600	Indicates that the local time is one hour (3600 seconds) earlier than Coordinated Universal Time (UTC).
time-server 128.128.1.1	Indicates the address of the time server for the network.

show async status

To display the status of activity on all lines configured for asynchronous support, use the **show async status** privileged EXEC command.

show async status

Syntax Description

This command has no arguments or keywords.

Command Mode

Privileged EXEC

Usage Guidelines

The display resulting from this command shows all asynchronous sessions, whether they are using SLIP or PPP encapsulation.

Sample Display

The following is sample output from the **show async status** command:

```
router# show async status

Async protocol statistics:
  Rcvd: 5448 packets, 7682760 bytes
        1 format errors, 0 checksum errors, 0 overrun, 0 no buffer
  Sent: 5455 packets, 7682676 bytes, 0 dropped


 Tty           Local          Remote Qd InPack OutPac Inerr  Drops  MTU Qsz
   1     192.31.7.84         Dynamic  0      0      0     0      0 1500  10
*  3     192.31.7.98            None  0   5448   5455     1      0 1500  10
```

Table 9-4 describes significant fields shown in the display.

Table 9-4 Show Async Status Display Field Descriptions

Field	Description
Rcvd:	Statistics on packets received.
5448 packets	Packets received.
7682760 bytes	Total number of bytes.
1 format errors	Spurious characters received when a packet start delimiter is expected.
0 checksum errors	Count of checksum errors.
0 overrun	Number of giants received.
0 no buffer	Number of packets received when no buffer was available.
Sent	Statistics on packets sent.
5455 packets	Packets sent.
7682676 bytes	Total number of bytes.
0 dropped	Number of packets dropped.

Field	Description
Tty	Line number.
*	Line currently in use.
Local	Local IP address on the link.
Remote	Remote IP address on the link; “Dynamic” indicates that a remote address is allowed but has not been specified; “None” indicates that no remote address is assigned or being used.
Qd	Number of packets on hold queue (Qsz is the maximum).
InPack	Number of packets received.
OutPac	Number of packets sent.
Inerr	Number of total input errors; sum of format errors, checksum errors, overruns and no buffers.
Drops	Number of packets received that would not fit on the hold queue.
MTU	Current maximum transmission unit size.
Qsz	Current output hold queue size.

show line

Use the **show line** privileged EXEC command to display connection status for a line running in asynchronous mode.

show line [*line-number*]

Syntax Description

line-number	(Optional) Particular line about which information will be displayed. If you do not specify a line number, information about all lines is displayed.

Command Mode

EXEC

Sample Display

The following is sample output from the **show line** command:

```
router> show line

 Tty Typ    Tx/Rx    A Modem  Roty AccO AccI  Uses    Noise   Overruns
*  0 CTY             -    -       -    -    -     0        0        0/0
A  1 TTY  9600/9600  -    -       -    -    1     0        0        0/0
   2 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   3 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   4 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   5 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   6 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   7 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   8 TTY  9600/9600  -    -       -    -    -     0        0        0/0
   9 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  10 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  11 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  12 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  13 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  14 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  15 TTY  9600/9600  -    -       -    -    -     0        0        0/0
  16 TTY  9600/9600  -    -       -    -    -     0        0        0/0
* 17 VTY  9600/9600  -    -       -    -    -    18        0        0/0
```

Table 9-5 describes significant fields shown in the display.

Table 9-5 Show Line Field Descriptions

Tasks	Descriptions
(first character in line)	The field preceding the number in the Tty field can be blank or contain one of the following characters: * The line is currently active, running a terminal-oriented protocol. A The line is currently active in asynchronous mode. I The line is free and can be used for asynchronous modes because it is configured for **async mode interactive.**
Tty	Indicates the absolute line number of the specified line.

Tasks	Descriptions
Typ	Type of line. Possible values follow: CTY—Console AUX—Auxiliary port TTY—Asynchronous terminal port VTY—Virtual terminal LPT—Parallel printer
Tx/Rx	Transmit rate of the line (baud)/receive rate of the line (baud).
A	Indicates whether or not autobaud has been configured for the line. A value of F indicates that autobaud has been configured; a hyphen (-) indicates that it has not been configured for the line.
Modem	Types of modem signal that has been configured for the line. Possible values include the following: callin callout cts-req DTR-Act inout RIisCD
Roty	Rotary Group configured for this line.
AccO	Output access list number configured for the specified line.
AccI	Input access list number configured for the specified line.
Uses	Number of connections established to or from this line since the system was restarted.
Noise	Number of times noise has been detected on the line since the system restarted.
Overruns	Hardware (UART) overruns/software buffer overflows, both defined as the number of overruns or overflows that have occurred on the specified line since the system was restarted. Hardware overruns are buffer overruns; the UART chip has received bits from the software faster than it can process them. A software overflow occurs when the software has received bits from the hardware faster than it can process them.

The following is sample output from the **show line** command when a line is specified:

```
router> show line 1

 Tty Typ     Tx/Rx    A Modem  Roty AccO AccI  Uses    Noise Overruns
   1 TTY  9600/9600  -    -      -    -   10     0        0        0

Line 1, Location: "charnel console", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600, no parity, 2 stopbits, 8 databits
Status: Ready, Hardware XON/XOFF
Capabilities: none
Modem state: Ready
Special Chars: Escape  Hold  Stop  Start  Disconnect  Activation
                ^^x    none   -     -       none
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
               0:10:00        never         0:00:15      not imp   not set
Session limit is not set.
Allowed transports are telnet lat rlogin.  Preferred is lat
```

```
No output characters are padded
Characters causing immediate data dispatching:
   Char    ASCII
Group codes:    0
```

Related Commands

async dynamic address
async dynamic routing
ip tcp header-compression

vty-async

To configure all virtual terminal lines on a router to support asynchronous protocol features, use the **vty-async** global configuration command. Use the **no vty-async** command to disable asynchronous protocol features on virtual terminal lines.

vty-async
no vty-async

Syntax Description

This command has no arguments or keywords.

Default

Asynchronous protocol features are not enabled by default on virtual terminal lines.

Command Mode

Global configuration

Usage Guidelines

The **vty-async** command extends asynchronous protocol features from physical asynchronous interfaces to virtual terminal lines. Normally, SLIP and PPP can function only on asynchronous interfaces, not on virtual terminal lines. However, extending asynchronous functionality to virtual terminal lines permits you to run SLIP and PPP on these *virtual asynchronous interfaces*. One practical benefit is the ability to tunnel SLIP and PPP over X.25 PAD, thus extending remote node capability into the X.25 area. You can also tunnel SLIP and PPP over Telnet or LAT on virtual terminal lines. When tunneling SLIP and PPP over X.25, LAT, or Telnet, you do so by using the protocol translation feature in the IOS software.

To tunnel SLIP or PPP inside X.25, LAT, or Telnet, you can use two-step protocol translation or one-step protocol translation, as follows:

- If you are tunnelling SLIP or PPP using the two-step method, you need to first enter the **vty-async** command on the router. Next, you perform two-step translation. For more information about two-step protocol translation, refer to the protocol translation chapter in the *Cisco Access Connection Guide*.
- If you are tunnelling SLIP or PPP using the one-step method, you do not need to enter the **vty-async** command. Instead, you would issue the **translate** command with the SLIP or PPP keywords, because the **translate** command automatically enables asynchronous protocol features on virtual terminal lines. For more information about protocol translation, refer to the "Configuring Protocol Translation Sessions" chapter in this publication. For more information about using the **translate** command with the SLIP or PPP keywords, refer to the "Protocol Translation Session Commands" chapter in this publication.

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

On a Cisco 3000, Cisco 4000, or Cisco 4500, you can create up to 180 protocol translation sessions, whether or not routing is enabled. Increase the number of virtual terminal lines using the **line vty** command.

Example

```
vty-async
```

Related Commands

A dagger (†) indicates that the command is documented in another chapter. Two daggers (††) indicate that the command is documented in the *Cisco Access Connection Guide*.

ppp ††
slip ††
translate†

vty-async dynamic-routing

To enable dynamic routing on all virtual asynchronous interfaces, use the **vty-async dynamic-routing** global configuration command. Use the **no vty-async** command to disable asynchronous protocol features on virtual terminal lines and, therefore, disable routing on virtual terminal lines.

vty-async dynamic-routing
no vty-async

Syntax Description

This command has no arguments or keywords.

Default

Dynamic routing is not enabled on virtual asynchronous interfaces.

Command Mode

Global configuration

Usage Guidelines

This feature enables IP routing on virtual asynchronous interfaces. When you issue this command and a user later makes a connection to another host using SLIP or PPP, the user must specify /**routing** at the SLIP or PPP command line.

If you had not previously entered the **vty-async** command, the **vty-async dynamic-routing** command creates virtual asynchronous interfaces on the router, then enables dynamic routing on them.

Command Mode

Global Configuration.

Example

The following command enables dynamic routing on virtual asynchronous interfaces.

```
vty-async dynamic-routing
```

Related Command

async dynamic routing

vty-async header-compression

To compress the headers of all TCP packets on virtual asynchronous interfaces, use the **vty-async header-compression** global configuration command. Use the **no vty-async** command to disable virtual asynchronous interfaces and header compression.

vty-async header-compression [**passive**]
no vty-async

Syntax Description

passive	(Optional) Specifies that outgoing packets to be compressed only if TCP incoming packets on the same virtual asynchronous interface are compressed. For SLIP, if you do not specify this option, the router will compress all traffic. The default is no compression. For PPP, the IOS software always negotiates header compression.

Default

Header compression is not enabled on virtual asynchronous interfaces.

Command Mode

Global configuration

Usage Guidelines

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

Example

The following example compresses outgoing TCP packets on virtual asynchronous interfaces only if incoming TCP packets are compressed:

```
vty-async header-compression passive
```

Related Command

async dynamic routing

vty-async keepalive

To change the frequency of keepalive packets on all virtual asynchronous interfaces, use the **vty-async keepalive** global configuration command. Use the **no vty-async** command to disable asynchronous protocol features on virtual terminal lines, or the **vty-async keepalive 0** command to disable keepalive packets on virtual terminal lines.

vty-async keepalive *seconds*
no vty-async
vty-async keepalive 0

Syntax Description

seconds	The frequency, in seconds, with which the IOS software sends keepalive messages to the other end of a virtual asynchronous interface. To disable keepalives, use a value of 0. The active keepalive interval is 1 through 32767 seconds.

Default

10 seconds

Command Mode

Global configuration

Usage Guidelines

Use this command to change the frequency of keepalive updates on virtual asynchronous interfaces from the default of 10, or to disable keepalive updates.

A connection is declared down after three update intervals have passed without receiving a keepalive packet.

Examples

In the following example, the keepalive interval is set to 30 seconds.

```
vty-async keepalive 30
```

In the following example, the keepalive interval is set to 0 (off), and the sample output for **write terminal** is shown.

```
vty-async keepalive 0
...
router# write terminal
no vty-async keepalive
```

Related Command

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

keepalive†

vty-async mtu

To set the maximum transmission unit (MTU) size on virtual asynchronous interfaces, use the **vty-async mtu** global configuration command. Use the **no vty-async** command to disable asynchronous protocol features on virtual terminal lines.

vty-async mtu *bytes*
no vty-async

Syntax Description

bytes	MTU size of IP packets that the virtual asynchronous interface can support. The default MTU is 1500 bytes, the minimum MTU is 64 bytes, and the maximum is 1000000 bytes.

Default

1500 bytes

Command Mode

Global configuration

Usage Guidelines

Use this command to modify the maximum transmission unit (MTU) for packets on a virtual asynchronous interfaces. You might want to change to a smaller MTU size for IP packets transmitted on a virtual terminal line configured for asynchronous functions for any of the following reasons:

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

Do not change the MTU size unless the SLIP or PPP implementation running on the host at the other end of the virtual asynchronous interface supports reassembly of IP fragments. Because each fragment occupies a spot in the output queue, it might also be necessary to increase the size of the SLIP or PPP hold queue, if your MTU size is such that you might have a high amount of packet fragments in the output queue.

Example

The following example sets the MTU for IP packets to 256 bytes:

```
vty-async mtu 256
```

Related Command

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

mtu†

vty-async ppp authentication

To enable PPP authentication on virtual asynchronous interfaces, use the **vty-async ppp authentication** {**chap** | **pap**} global configuration command. Use the **no vty-async** command to globally disable asynchronous protocol features on virtual terminal lines, or the no **vty-async ppp authentication** {**chap** | **pap**} command to disable PPP authentication.

vty-async ppp authentication {**chap** | **pap**}
no vty-async
no vty-async ppp authentication {**chap** | **pap**}

Syntax Description

chap	Enable CHAP on all virtual asynchronous interfaces on the router.
pap	Enable PAP on all virtual asynchronous interfaces on the router.

Default

No CHAP or PAP authentication for PPP.

Command Mode

Global configuration

Usage Guidelines

This command configures the virtual asynchronous interface to authenticate either CHAP or PAP while running PPP. Once you have enabled CHAP or PAP, the local router requires a password from remote devices. If the remote device does not support CHAP or PAP, no traffic will be passed to that device.

Example

The following example enables CHAP authentication for PPP sessions on virtual asynchronous interfaces:

```
vty-async authentication ppp chap
```

Related Commands

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

ppp authentication chap †
ppp authentication pap †
ppp use-tacacs †
vty-async ppp use-tacacs

vty-async ppp use-tacacs

To enable TACACS authentication for PPP on virtual asynchronous interfaces, use the **vty-async ppp** global configuration command. Use the **no vty-async** command to disable virtual asynchronous interfaces, or the no **vty-async use-tacacs** command to disable TACACS authentication on virtual asynchronous interfaces.

vty-async ppp use-tacacs
no vty-async
no vty-async ppp use-tacacs

Syntax Description

This command has no arguments or keywords.

Default

TACACS for PPP is disabled.

Command Mode

Global configuration

Usage Guidelines

Use this command only when you have set up an extended TACACS server. This command requires the extended TACACS server.

Once you have enabled TACACS, the local router requires a password from remote devices.

This feature is useful when integrating TACACS with other authentication systems that require a clear-text version of a user's password. Such systems include one-time password systems, token card systems, and others.

If the username and password are contained in the CHAP password, then the CHAP secret is not used by the router. Because most PPP clients require that a secret be specified, you can use any arbitrary string; the router ignores it.

You cannot enable TACACS authentication for SLIP on asynchronous or virtual asynchronous interfaces.

Example

The example enables TACACS authentication for PPP sessions:

```
vty-async ppp use-tacacs
```

Related Commands

A dagger (†) indicates that the command is documented in the *Router Products Command Reference* publication.

ppp use-tacacs †
vty-async ppp authentication {**chap** | **pap**}

CHAPTER 10

XRemote Configuration Commands

The X Window System, also called X, is a network-based graphics window system originally developed for workstations running UNIX. Cisco Systems developed an XRemote application that allows the XRemote capabilities of X terminals to run on the router. Information in this chapter will help you understand the X Windows system and how to configure your router to support XRemote connections.

For XRemote configuration information and examples, refer to the chapter "Configuring XRemote" earlier in this publication. For information about establishing XRemote connections, refer to the *Cisco Access Connection Guide*.

show xremote

To display XRemote connections and monitor XRemote traffic through the router, use the **show xremote** EXEC command.

show xremote

Syntax Description

This command has no arguments or keywords.

Command Mode

EXEC

Usage Guidelines

The **show xremote** command displays XRemote parameters applied to the entire system, as well as statistics that are pulled for all active XRemote connections.

Sample Display

The following is sample output from the **show xremote** command when XRemote is enabled on a router and XRemote sessions are active:

```
pt# show xremote

XRemote server-wide parameters:
  Font buffersize:     72000              Font retries: 3
  Font memory errors:  0

TFTP font load statistics for host 131.108.1.111:
  Bytes read:          2697239            Files read: 258
  Network errors:      4                  File errors: 0

LAT font load statistics for service WHEEL, incarnation 5:
 Bytes read            182401             Files read: 14
 Protocol errors:      0                  Insufficient memory: 0

XRemote statistics for tty2:
  Current clients:     9                  Total clients: 17
  Requesting client:   5                  Current request size: 0
  Replying client:     6                  Current reply size: 0
  XDM state:           10                 Next timeout: 172460
  Retransmit counter:  0                  Local UDP port: 53616
  Keepalive dormancy:  180                Session id: 94
  Unread input:        0                  Unwritten output: 0
  Input buffer size:   1024               Output buffer size: 108
  Protocol version:    2                  Line state: Connected
  Transmit packets:    50768              Receive packets: 49444
  Transmit errors:     0                  Receive errors: 37
  Retransmissions:     403                Receive out of sequence: 76
  Round trip time:     383                Retransmit interval: 766
  Transmit window:     7                  Receive window: 7
  Transmit next:       6                  Receive next: 3
  Transmit unacked:    6                  Receive unacked: 0

  Connection 0 - TCP connection from 131.108.1.55  [Display Manager]
    Client state:       CS_ACTIVE         Byte order: MSBfirst
    Unread input:       0                 Unwritten output: 0
```

```
   Input buffer size:  1024             Output buffer size: 1024

Connection 1 - LAT connection from WHEEL
   Client state:       CS_ACTIVE        Byte order: LSBfirst
   Unread input:       0                Unwritten output: 0
   Input buffer size:  1024             Output buffer size: 1024
```

Table 10-1 describes significant fields shown in the display.

Table 10-1 Show XRemote Field Descriptions

Field	Description
XRemote server-wide parameters	This section displays XRemote parameters that apply to the router.
Font buffersize	XRemote font buffer size that was specified with the **xremote tftp buffersize** global configuration command.
Font retries	Number of retries the font loader (host) will attempt before declaring an error condition.
Font memory errors	Number of font memory error conditions that have been declared for the router.
TFTP font load statistics for host 131.108.1.111	This section displays XRemote statistics for fonts that have been loaded from a TFTP font server at the IP address shown.
Bytes read	Number of bytes the host read in order to load the fonts.
Files read	Number of files the host read in order to load the fonts.
XRemote statistics for tty2	This section displays XRemote for the specified line.
Current clients	Number of clients using this line for active XRemote sessions.
Total clients	Includes the number of clients using this line for active XRemote sessions.
Requesting client	Number of clients requesting XRemote service.
Retransmit counter	Number of times that XRemote connection requests were retransmitted.
Local UDP port	Number assigned to the local UDP port.
Keepalive dormancy	Amount of time between keepalive messages.

show xremote line

To list XRemote connections and monitor XRemote traffic, use the **show xremote line** EXEC command.

show xremote line *number*

Syntax Description

number	A decimal value representing the number of virtual terminal lines about which to display information.

Command Mode

EXEC

Sample Display

The following is sample output from the **show xremote line** command (line 3 is specified) when XRemote is enabled and XRemote sessions are active. Only information specific to an individual terminal line is provided.

```
router# show xremote line 3

Xremote statistics for tty3:
  Current clients:     11                Total clients: 19
  Requesting client:   10                Current request size: 0
  Replying client:     10                Current reply size: 0
  XDM state:           10                Next timeout: 173304
  Retransmit counter:  0                 Local UDP port: 28384
  Keepalive dormancy:  180               Session id: 29
  Unread input:        0                 Unwritten output: 0
  Input buffer size:   1024              Output buffer size: 108
  Protocol version:    2                 Line state: Connected
  Transmit packets:    28875             Receive packets: 18644
  Transmit errors:     0                 Receive errors: 13
  Retransmissions:     53                Receive out of sequence: 41
  Round trip time:     384               Retransmit interval: 768
  Transmit window:     7                 Receive window: 7
  Transmit next:       2                 Receive next: 7
  Transmit unacked:    2                 Receive unacked: 0

  Connection 0 - TCP connection from 131.108.1.27  [Display Manager]
    Client state:       CS_ACTIVE        Byte order: MSBfirst
    Unread input:       0                Unwritten output: 0
    Input buffer size:  1024             Output buffer size: 1024

  Connection 1 - TCP connection from 131.108.1.27
    Client state:       CS_ACTIVE        Byte order: MSBfirst
    Unread input:       0                Unwritten output: 0
    Input buffer size:  1024             Output buffer size: 1024

  Connection 2 - TCP connection from 131.108.1.27
    Client state:       CS_ACTIVE        Byte order: MSBfirst
    Unread input:       0                Unwritten output: 0
    Input buffer size:  1024             Output buffer size: 1024
```

Table 10-1 earlier in this chapter describes the fields shown in the display.

xremote tftp buffersize

To change the buffer size used for loading font files, use the **xremote tftp buffersize** global configuration command. To restore the buffer size to the default value, use the **no** form of this command.

xremote tftp buffersize [*buffer-size*]
no xremote tftp buffersize

Syntax Description

buffer-size	(Optional) Buffer size in bytes. This is decimal number in the range 4096 to 70000 bytes.

Default

70000 bytes

Command Mode

Global configuration

Usage Guidelines

When an X terminal requests that a font file be loaded, the router first must load the font file into an internal buffer before passing it to the X terminal. The default value of 70000 bytes is adequate for most font files, but the size can be increased as necessary for nonstandard font files.

The buffer size can be set as low as 4096 bytes and as large as the available memory on the router will allow. If you are using LAT font access, you should not lower the buffer size below the default, because the font directory for all of the LAT fonts (created internally) requires 70000 bytes.

This command applies to both TFTP and LAT font access.

Example

The following example sets the buffer size to 20000 bytes:

```
xremote tftp buffersize 20000
```

xremote tftp host

To add a specific TFTP font server as a source of fonts for the terminal, use the **xremote tftp host** global configuration command. To remove a font server from the list, use the **no** form of this command.

xremote tftp host *hostname*
no xremote tftp host *hostname*

Syntax Description

hostname	IP address or name of the host containing fonts.

Default

No TFTP font server specified.

Command Mode

Global configuration

Usage Guidelines

Each time a new host name is entered, the list on the router is updated. Font servers are queried in the order of their definition when the X terminal requests a font.

Examples

The following example sets the host IBM-1 as an XRemote TFTP font server:

```
xremote tftp host IBM-1
```

The following example sets the host with IP address 1.0.0.7 as an XRemote TFTP font server:

```
xremote tftp host 1.0.0.7
```

xremote tftp retries

To specify the number of retries the font loader will attempt before declaring an error condition, use the **xremote tftp retries** global configuration command. To restore the default number of retries, use the **no** form of this command.

xremote tftp retries [*retries*]
no xremote tftp retries

Syntax Description

retries	(Optional) Number of retries. This is a decimal number in the range 1 to 15.

Default

3 retries

Command Mode

Global configuration

Usage Guidelines

Under certain conditions, you might need to increase the number of retries, particularly if the font servers are known to be heavily loaded.

Example

The following example sets the font loader retries to 5:

```
xremote tftp retries 5
```

CHAPTER 11

Protocol Translation Session Commands

The protocol translation software attempts to provide transparent translation between systems running disparate protocols. The software fully supports two-way virtual terminal protocol translation between nodes running X.25, Local Area Transport (LAT), SLIP or PPP, and Telnet, a remote terminal protocol that is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite.

To provide fully transparent protocol conversion, the router masquerades as two or more hosts on the same network. When a connection is made to the router, the router determines which host the connection is for and what protocol that host is using. The router then establishes a new network connection using the networking protocol required by that host. This network connection is more efficient and allows the router to act upon greater knowledge of the protocols in use because the router acts as a network connection rather than a terminal.

This chapter describes the commands used to configure protocol translation connections. For protocol translation configuration information and examples, refer to the chapter "Configuring Protocol Translation Sessions" earlier in this publication.

For information about protocol translation EXEC commands, such as the **pad** EXEC command, refer to the *Cisco Access Connection Guide.*

show translate

To view translation sessions that have been configured on a router, enter the following command in privileged EXEC mode:

show translate

Syntax Description

This command has no arguments or keywords.

Command Mode

Privileged EXEC

Usage Guidelines

The display from this command shows each translation session set up on the router. It shows the incoming device and virtual terminal protocol as well as the outgoing device and protocol.

Sample Display

The **show translate** output in this sample display is based on the following **translate** command configured on the router:

```
translate lat ramble ppp 172.21.10.10 0 keepalive

CS# show translate

Translate From: LAT ramble
          To:   PPP 172.21.10.10  keepalive 0
          1/1 users active, 1 peak, 1 total, 0 failures
```

Table 11-1 describes fields shown in the display.

Table 11-1 Show Translate Field Descriptions

Field	Description
Translate From: LAT ramble	The virtual terminal protocol (LAT) and hostname (ramble) of the incoming device.
To: PPP 172.21.10.10	The virtual terminal protocol (PPP) and IP address (172.21.10.10) of the outgoing device.
keepalive 0	Indicates that keepalive updates have been disabled for the current translation session.
1/1 users active	Number of users active over the total number of users.
1 peak	Maximum number of translate sessions up at any given time.
1 total	Total number of translation sessions.
0 failures	Number of failed translation attempts resulting from this configuration.

translate

To automatically convert incoming LAT, TCP, or X.25 requests for connections to a specified destination address or host name to the specified outgoing connection type, use the **translate** global configuration command.

translate *protocol incoming-address* [*in-options*] *protocol outgoing-address* [*out-options*] [*global-options*]

Syntax Description

protocol incoming-address
protocol outgoing-address

Name of a protocol followed by a service name or address. These arguments can have the following values:

- **lat** *service-name*—LAT and a LAT service name. The application of *service-name* can differ, depending on whether it is being used on the incoming or the outgoing portion of the command. When used on the incoming portion, *service-name* is the name of the service that users specify when trying to make a translated connection. This name can match the name of final destination resource, but this is not required. This can be useful when making remote translated connections.
- **x25** *X.121-address*—X.25 and an X.121 address. The X.121 address must conform to specifications provided in the *CCITT 1984 Red Book*. This number generally consists of a portion that is administered by the PDN and a portion that is locally assigned. You must be sure that the numbers that you assign are in agreement with addresses assigned to you by the X.25 service provider. The X.121 addresses will generally be subaddresses of the X.121 address of the X.25 network interface. Typically, the interface address will be a 12-digit number. Any additional digits are interpreted as a subaddress. The PDN still routes these calls to the interface, and the router itself is responsible for dealing with the extra digits appropriately.
- **tcp** *ip-address*—TCP/IP Telnet and a standard IP address or host name. The argument *ip-address* is a standard, four-part dotted decimal IP address or the name of an IP host that can be resolved by the Domain Name System (DNS) or explicit specification in an **ip host** command.
- **slip** *ip-address*—The argument *ip-address* is a standard, four-part dotted decimal IP address or the name of an IP host that can be resolved by the Domain Name System (DNS). The **slip** argument applies only to outgoing connections; SLIP is not supported on incoming protocol translation connections.

- **ppp** *ip-address*—The argument *ip-address* is a standard, four-part dotted decimal IP address or the name of an IP host that can be resolved by the Domain Name System (DNS). The **ppp** argument applies only to outgoing connections; PPP is not supported for incoming protocol translation connections.

in-options
out-options

(Optional) Incoming and outgoing connection request options. These arguments can have the following values:

For LAT translation options:

- **node** *node-name*—Outgoing connection request which connects to the specified node (*node-name*) that offers a service. By default, the connection is made to the highest-rated node that offers the service.
- **port** *port-name*—Destination LAT port name (*port-name*) in the format of the remote system. This parameter is usually ignored in most timesharing systems, but is used by terminal servers that offer reverse-LAT services. Outgoing connection request only.
- **unadvertised**—Prevents service advertisements from being broadcast to the network. This can be useful, for example, when you define translations for many printers, and you do not want these services advertised to other LAT terminal servers. (VMS systems will be able to connect to the service even though it is not advertised.)

For X.25 translation options:

- **cud** *c-u-data*—Sends the specified Call User Data (CUD) text (*c-u-data*) as part of an outgoing call request after the protocol identification bytes.
- **profile** *profile*—Sets the X.3 PAD parameters as defined in the profile created by the **x29 profile** command.
- **reverse**—Provides reverse charging for X.25 on a per-call rather than a per-interface basis. Requests reverse charges on a specified X.121 address, even if the serial interface is not configured to request reverse charge calls. This is an outgoing option only.
- **accept-reverse**—Accepts reverse charged calls on an X.121 address even if the serial interface is not configured to accept reverse charged calls. This is an incoming option only.

- **printer**—Supports LAT and TCP printing over an X.25 network among multiple sites. Provides an "interlock mechanism" between the acceptance of an incoming X.25 connection and the opening of an outgoing LAT or TCP connection. The option causes the router to delay the call confirmation of an incoming X.25 call request until the outgoing protocol connection (to TCP or LAT) has been successfully established. An unsuccessful outgoing connection attempt results in the incoming X.25 connection to the router being refused, rather than being confirmed and then cleared, which is the default behavior. Note that using this option will force the global option *quiet* to be applied to the translation. Incoming connection request only.

- **pvc** *number*—Specifies that the incoming connection (identified by the argument *number*) is actually a permanent virtual circuit (PVC). Incoming connection request only.

For Telnet TCP translation options:

- **port** *number*—For incoming connections, number of the port to match. The default is port 0 (any port). For outgoing connections, number of the port to use. The default is port 23 (Telnet).

- **binary**—Negotiates Telnet binary mode on the Telnet connection. (This was the default in previous versions of the protocol translation software and is set automatically when you enter at **translate** command in the old format.)

- **stream**—Performs stream processing, which enables a raw TCP stream with no Telnet control sequences. A stream connection does not process or generate any Telnet options, and prevents Telnet processing of the data stream as well. This option might be useful for connections to ports running UUCP or other non-Telnet protocols, or to ports connected to printers. For ports connected to printers using Telnet, the stream option prevents some of usual problems associated with using Telnet for printers, such as strange things happening to bare carriage returns or line feeds and echoing of data back to VMS systems.

- **printer**—Supports LAT and X.25 printing over a TCP network among multiple sites. Causes the protocol translation software to delay the completion of an incoming Telnet connection until after the outgoing protocol connection (to LAT or X.25) has been successfully established. An unsuccessful outgoing connection attempt results in the TCP connection to the router being refused, rather than being accepted and then closed, which is the default behavior. Note that using this option will force the global option *quiet* to be applied to the translation.

For SLIP and PPP translation options: These variables apply to *out-options* only; SLIP and PPP are not supported on incoming protocol translation connections

- **headercompressed** [**passive**]—Implements header compression on IP packets only. The option **passive** permits compression on outgoing packets only if incoming TCP packets on the same virtual asynchronous interface are compressed. The default (without the **passive** option) permits compression on all traffic.
- **routing**—Permits routing updates between connections. This option is required if the destination device is not on a subnet connected to one of the interfaces on the router running protocol translation software.
- **keepalive** *number-of-seconds*—Permits you to specify the interval at which keepalive packets are sent on SLIP and PPP lines. By default, keepalives are enabled and are sent every 10 seconds. To disable keepalives, use a value of 0. The active keepalive interval is 1 through 32767 seconds.
- **mtu** *bytes*—Permits you to change the maximum transmission unit (MTU) of packets that the virtual asynchronous interface supports. The default MTU is 1500 bytes on a virtual asynchronous interface. The acceptable range is 64 through 1000000 bytes.
- **ppp authentication** {**chap** | **pap**}—Use CHAP or PAP authentication for PPP on virtual asynchronous interfaces. Refer to the "Configuring SLIP and PPP" chapter in the *Router Products Configuration Guide Addendum* for more information about enabling authentication on virtual asynchronous interfaces.
- **ppp use-tacacs**—Enable TACACS authentication for CHAP or PAP on virtual asynchronous interfaces (for PPP only; TACACS authentication is not supported for SLIP).

global-options

(Optional) Translation options that can be used by any connection type. It can be one or more of the following:

- **access-class** *number*—Allows the incoming call to be used by source hosts that match the access list parameters. The argument *number* is the number (integer) previously assigned to an access list. This feature is supported only for incoming TCP and X.25 connections.
- **max-users** *number*—Limits the number of simultaneous users of the translation to *number* (an integer you specify).
- **local**—Allows Telnet protocol negotiations to *not* be translated.

- **login**—Requires that the user log in before the outgoing connection is made. This type of login is specified on the virtual terminal lines using the **login** command.
- **quiet**—Suppresses printing of user-information messages.
- **swap**—Allows X.3 parameters to be set on the router by the host originating the X.25 call, or by an X.29 profile. This allows incoming and outgoing X.25 connections to be swapped so that the router is treated like a PAD when it accepts a call. By default, the router behaves like a PAD for calls that it initiates, and behaves like an X.25 host for calls it accepts. The **swap** keyword allows connections from an X.25 host that wants to connect to the router, and then treats it like a PAD. For X.25-to-TCP translations only.

Default

No default translation parameters

Command Mode

Global configuration

Usage Guidelines

Table 11-2 provides a visual aid for understanding how to use the **translate** command. As the table illustrates, you define the protocol translation connections—both incoming and outgoing—by choosing a protocol keyword and supplying the appropriate address or service name. The protocol connection information is followed by optional features for that connection, also as appropriate. For example, the **binary** option is only appropriate with TCP/IP connections. The global options, in general, apply to all the connection types, but there are exceptions. The **swap** keyword, for example, is for X.25 to TCP translations only. See the examples for more explanations on how to enter this command.

Table 11-2 Translate Command Options

	Incoming Protocol	Options	Outgoing Protocol	Options	Global Options
translate	*protocol incoming-address*	[*in-options*]	*protocol outgoing-address*	[*out-options*]	[*global-options*]
	lat *service-name*	**unadvertised**	**lat** *service-name*	**node** *node-name*	**access-class** *number*
				port *port-name*	**max-users** *number*
					local
					login
	x25 *x.121-address*	**cud** *c-u-data*	**x25** *x.121-address*	**cud** *c-u-data*	**quiet**
		profile *profile*		**profile** *profile*	**swap**
		accept-reverse		**reverse**	
		printer			

Incoming Protocol	Options	Outgoing Protocol	Options	Global Options
	pvc *number*			
tcp *ip-address*	**port** *number*	**tcp** *ip-address*	**port** *number*	
	binary			
	stream			
	printer			
		slip *ip-address*	**headercompressed** [**passive**]	
			routing	
			keepalive *number-of-seconds*	
			mtu *bytes*	
		ppp *ip-address*	**headercompressed** [**passive**]	
			routing	
			keepalive *number-of-seconds*	
			mtu *bytes*	
			ppp authentication {**pap** | **chap**}	
			ppp use-tacacs	

Examples

The following example illustrates a simple X.25 to TCP translation command. Packets coming in X.25 address 652365123 arrive via PVC 1 and are translated to TCP packets and transmitted out IP address 131.108.1.1.

```
translate x25 652365123 pvc 1 tcp 131.108.1.1
          incoming      option outgoing
```

The following example illustrates incoming LAT to outgoing TCP translations. The **unadvertised** keyword prevents broadcast of service advertisements to other servers. Outgoing translated packets are transmitted out IP address rubble via TCP port 4005.

```
translate lat pt-printer1 unadvertised tcp rubble port 4005
          incoming        option       outgoing   option
```

The following example illustrates a more complex configuration that calls an X.29 profile and swaps the default PAD operation of the router to that of an X.25 host.

```
x29 profile fullpackets 2:0 3:0 4:100 7:21
translate x25 217536124 profile fullpackets tcp rubble port 4006 swap
          incoming      option              outgoing   option    global
```

The following example illustrates the use of the TCP incoming protocol option **printer** for an incoming TCP connection.

```
translate tcp 160.89.32.250 printer x25 5678
          incoming          option  outgoing
```

The following example illustrates the use of the X.25 incoming protocol option **printer** for an incoming X.25 connection.

```
translate x25 55555  printer tcp 131.108.1.1
          incoming   option  outgoing
```

The following example translates LAT on an incoming line to SLIP on an outgoing line. It uses header compression only if incoming TCP packets on the same interface are compressed.

```
translate lat rudolph slip 1.0.0.4 headercompressed
          incoming    outgoing     option
```

The following example translates X.25 packets to PPP. It enables routing updates between the two connections.

```
translate x25 12345678 ppp 1.0.0.2 routing
          incoming     outgoing    option
```

The following example first shows the command to disable keepalives on a PPP line, then shows sample output from the **show translate** command when keepalives have been turned off on the line.

```
translate lat ramble ppp 172.21.2.2 keepalive 0
.
.
.
router# show translate

Translate From: LAT ramble
          To:   PPP 172.21.2.2 no-keepalive
          1/0 users active, 1 peak, 1 total, 0 failures
```

x25 host

Use the **x25 host** global configuration command to define a static host name for address mapping. Use the **no x25 host** command to remove the host name.

x25 host *name x.121-address* [**cud** *call-user-data*]
no x25 host *name*

Syntax Description

name	Host name.
x.121-address	X.121 address.
cud *call-user-data*	(Optional) Specifies the Call User Data (CUD) field in the X.25 Call Request packet.

Default

No static address mapping is defined.

Command Mode

Global configuration

Usage Guidelines

This command permits you to map an X.121 address to an easily recognizable name. You can later use this host name instead of the X.121 address when you issue the **translate** command for X.25.

Examples

The following example specifies a static address mapping:

```
x25 host Willard 4085551212
```

The following example removes a static address mapping:

```
no x25 host Willard
```

The following example specifies static address mapping from the X.121 address 12345678 to the host name masala. It then uses the name masala in the **translate** command in place of the X.121 address when translating from the X.25 host to the PPP host with address 1.0.0.2.

```
x25 host masala 12345678
translate x25 masala ppp 1.0.0.2 routing
```

Related Command

translate

x29 access-list

To limit access to the router from certain X.25 hosts, use the **x29 access-list** global configuration command. To delete an entire access list, use the **no** form of this command.

x29 access-list *access-list-number* {**permit** | **deny**} *regular-expression*
no x29 access-list *access-list-number*

Syntax Description

access-list-number	Number of the access list. It can be a value between 1 and 199.
deny	Denies access and clears call requests immediately.
permit	Permits access to the router.
regular-expression	Usually the X.121 address, with or without regular expression pattern-matching characters, with which to compare for access.

Default

No default access list is defined.

Command Mode

Global configuration

Usage Guidelines

An access list can contain any number of access list items. The list are processed in the order in which you entered them, with the first match causing the permit or deny condition. If an X.121 address does not match any of the regular expression in the access list, access will be denied.

Access lists take advantage of the message field defined by Recommendation X.29, which describes procedures for exchanging data between two PADs or a PAD and a DTE device.

The UNIX-style regular expression characters allow for pattern matching of characters and character strings in the address. Various pattern-matching constructions are available that will allow many addresses to be matched by a single regular expressions. Refer to the appendix “X.3 PAD Parameters” later in this publication for more information.

Example

The following example permits connections to hosts with addresses beginning with the string 31370:

```
x29 access-list 2 permit ^31370
```

x29 profile

To create a PAD profile script for use by the **translate** command, use the **x29 profile** global configuration command.

x29 profile *name parameter*:*value* [*parameter*:*value*]

Syntax Description

name	Name of the PAD profile script.
parameter:*value*	(Optional) X.3 PAD parameter number and value separated by a colon. You can specify multiple parameter-value pairs.

Default

No default PAD profile script defined.

Command Mode

Global configuration

Usage Guidelines

When an X.25 connection is established, the router acts as if an X.29 SET PARAMETER packet had been sent containing the parameters and values set by the **x29 profile** command and sets the router accordingly.

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. The name *linemode* is used with the **translate** global configuration command to effect use of this script.

```
x29 profile linemode 2:1 3:2 15:1
```

Related Command

translate

Appendixes

APPENDIX A

References and Recommended Reading

Books and Periodicals

Apple Computer, Inc. *AppleTalk Network System Overview.* Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1989.

Apple Computer, Inc. *Planning and Managing AppleTalk Networks*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1991.

Black, U. *Data Networks: Concepts, Theory and Practice.* Englewood Cliffs, New Jersey: Prentice Hall; 1989.

Black, U. Physical Level Interfaces and Protocols. Los Alamitos, California: IEEE Computer Society Press; 1988.

Case, J.D., J.R. Davins, M.S. Fedor, and M.L. Schoffstall. "Network Management and the Design of SNMP." ConneXions: The Interoperability Report, Vol. 3: March 1989.

Case, J.D., J.R. Davins, M.S. Fedor, and M.L. Schoffstall. "Introduction to the Simple Gateway Monitoring Protocol." IEEE Network: March 1988.

Clark, W. "SNA Internetworking." *ConneXions: The Interoperability Report*, Vol. 6, No. 3: March 1992.

Coltun, R. "OSPF: An Internet Routing Protocol." *ConneXions: The Interoperability Report*, Vol. 3, No. 8: August 1989.

Comer, D.E. *Internetworking with TCP/IP: Principles, Protocols, and Architecture*, Vol. I, 2nd ed. Englewood Cliffs, New Jersey: Prentice Hall; 1991.

Davidson, J. *An Introduction to TCP/IP.* New York, New York: Springer-Verlag; 1992.

Ferrari, D. *Computer Systems Performance Evaluation*. Englewood Cliffs, New Jersey: Prentice Hall; 1978.

Garcia-Luna-Aceves, J.J. "Loop-Free Routing Using Diffusing Computations." Publication pending in *IEEE/ACM Transactions on Networking*, Vol. 1, No. 1, 1993.

Green, J.K. *Telecommunications*, 2nd ed. Homewood, Illinois: Business One Irwin; 1992.

Hagans, R. "Components of OSI: ES-IS Routing." *ConneXions: The Interoperability Report*, Vol. 3, No. 8: August 1989.

Hares, S. "Components of OSI: Inter-Domain Routing Protocol (IDRP)." *ConneXions: The Interoperability Report*, Vol. 6, No. 5: May 1992.

Jones, N.E.H. and D. Kosiur. *Macworld Networking Handbook*. San Mateo, California: IDG Books Worldwide, Inc.; 1992.

Joyce, S.T. and J.Q. Walker II. "Advanced Peer-to-Peer Networking (APPN): An Overview." *ConneXions: The Interoperability Report*, Vol. 6, No. 10: October 1992.

Kousky, K. "Bridging the Network Gap." *LAN Technology*, Vol. 6, No. 1: January 1990.

Leinwand, A. and K. Fang. *Network Management: A Practical Perspective*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1993.

Lippis, N. "The Internetwork Decade." *Data Communications*, Vol. 20, No. 14: October 1991.

Malamud, C. *Analyzing DECnet/OSI Phase V.* New York, New York: Van Nostrand Reinhold; 1991.

Malamud, C. *Analyzing Novell Networks*. New York, New York: Van Nostrand Reinhold; 1991.

Malamud, C. *Analyzing Sun Networks*. New York, New York: Van Nostrand Reinhold; 1991.

Martin, J. *SNA: IBM's Networking Solution*. Englewood Cliffs, New Jersey: Prentice Hall; 1987.

Martin, J., with K.K. Chapman and the ARBEN Group, Inc. *Local Area Networks. Architectures and Implementations*. Englewood Cliffs, New Jersey: Prentice Hall; 1989.

McNamara, J.E. *Local Area Networks*. Digital Press, Educational Services, Digital Equipment Corporation, 12 Crosby Drive, Bedford, MA 01730.

Medin, M. "The Great IGP Debate—Part Two: The Open Shortest Path First (OSPF) Routing Protocol." *ConneXions: The Interoperability Report*, Vol. 5, No. 10: October 1991.

Meijer, A. *Systems Network Architecture: A tutorial*. New York, New York: John Wiley & Sons, Inc.; 1987.

Miller, M.A. *LAN Protocol Handbook*. San Mateo, California: M&T Books; 1990.

Miller, M.A. *LAN Troubleshooting Handbook*. San Mateo, California: M&T Books; 1989.

O'Reilly, T. and G. Todino. *Managing UUCP and Usenet*, 10th ed. Sebastopol, California: O'Reilly & Associates, Inc.; 1992.

Perlman, R. *Interconnections: Bridges and Routers.* Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1992.

Perlman, R. and R. Callon. "The Great IGP Debate—Part One: IS-IS and Integrated Routing." *ConneXions: The Interoperability Report*, Vol. 5, No. 10: October 1991.

Rose, M.T. *The Open Book: A Practical Perspective on OSI*. Englewood Cliffs, New Jersey: Prentice Hall; 1990.

Rose, M.T. *The Simple Book: An Introduction to Management of TCP/IP-based Internets.* Englewood Cliffs, New Jersey: Prentice Hall; 1991.

Ross, F.E. "FDDI—A Tutorial." *IEEE Communications Magazine*, Vol. 24, No. 5: May 1986.

Schlar, S.K. *Inside X.25: A Manager's Guide*. New York, New York: McGraw-Hill, Inc.; 1990.

Schwartz, M. *Telecommunications Networks: Protocols, Modeling, and Analysis*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1987.

Sherman, K. *Data Communications: A User's Guide.* Englewood Cliffs, New Jersey: Prentice Hall; 1990.

Sidhu, G.S., R.F. Andrews, and A.B. Oppenheimer. *Inside AppleTalk*, 2nd ed. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1990.

Spragins, J.D. et al. *Telecommunications Protocols and Design*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1991.

Stallings, W. *Data and Computer Communications*. New York, New York: Macmillan Publishing Company; 1991.

Stallings, W. *Handbook of Computer-Communications Standards*, Vols. 1–3. Carmel, Indiana: Howard W. Sams, Inc.; 1990.

Stallings, W. *Local Networks*, 3rd ed. New York, New York: Macmillan Publishing Company; 1990.

Sunshine, C.A. (ed.). *Computer Network Architectures and Protocols*, 2nd ed. New York, New York: Plenum Press; 1989.

Tannenbaum, A.S. *Computer Networks*, 2nd ed. Englewood Cliffs, New Jersey: Prentice Hall; 1988.

Terplan, K. *Communication Networks Management*. Englewood Cliffs, New Jersey: Prentice Hall; 1992.

Tsuchiya, P. "Components of OSI: IS-IS Intra-Domain Routing." *ConneXions: The Interoperability Report*, Vol. 3, No. 8: August 1989.

Tsuchiya, P. "Components of OSI: Routing (An Overview)." *ConneXions: The Interoperability Report*, Vol. 3, No. 8: August 1989.

Zimmerman, H. "OSI Reference Model—The ISO Model of Architecture for Open Systems Interconnection." *IEEE Transactions on Communications* COM-28, No. 4: April 1980.

Technical Publications and Standards

Advanced Micro Devices. *The Supernet Family for FDDI*. Technical Manual Number 09779A. Sunnyvale, California; 1989.

———. *The Supernet Family for FDDI*. 1989 Data Book Number 09734C. Sunnyvale, California; 1989.

American National Standards Institute X3T9.5 Committee. *FDDI Station Management (SMT)*. Rev. 6.1; March 15, 1990.

———. Revised Text of ISO/DIS 8802/2 for the Second DIS Ballot, "Information Processing Systems—Local Area Networks." Part 2: Logical Link Control. 1987-01-14.

——— T1.606. Integrated Services Digital Network (ISDN)—Architectural Framework and Service Description for Frame-Relaying Bearer Service. 1990.

——— T1.617. Integrated Services Digital Network (ISDN)—Signaling Specification for Frame Relay Bearer Service for Digital Subscriber Signaling System Number 1 (DSS1). 1991.

——— T1.618. Integrated Services Digital Network (ISDN)—Core Aspects of Frame Protocol for Use with Frame Relay Bearer Service. 1991.

ATM Data Exchange Interface (DXI) Specification, Version 1.0. Document ATM_FORUM/93-590R1; August 4,1993.

Banyan Systems, Inc. *VINES Protocol Definition*. DA254-00, Rev. 1.0. Westboro, Massachusetts; February 1990.

Bellcore. *Generic System Requirements in Support of a Switched Multi-Megabit Data Service*. Technical Advisory, TA-TSY-000772; October 1989.

———. *Local Access System Generic Requirements, Objectives, and Interface Support of Switched Multi-Megabit Data Service*. Technical Advisory TA-TSY-000773, Issue 1; December 1985.

———. *Switched Multi-Megabit Data Service (SMDS) Operations Technology Network Element Generic Requirements*. Technical Advisory TA-TSY-000774.

Chapman, J.T. and M. Halabi. *HSSI: High-Speed Serial Interface Design Specification*. Menlo Park, California and Santa Clara, California: Cisco Systems and T3Plus Networking, Inc.; 1990.

Consultative Committee for International Telegraph and Telephone. *CCITT Data Communications Networks—Services and Facilities, Terminal Equipment and Interfaces, Recommendations X.1–X.29.* Yellow Book, Vol. VIII, Fascicle VIII.2; 1980.

———. *CCITT Data Communications Networks—Interfaces, Recommendations X.20–X.32.* Red Book, Vol. VIII, Fascicle VIII.3; 1984.

DDN Protocol Handbook. Four volumes; 1989.

Defense Communications Agency. *Defense Data Network X.25 Host Interface Specificiation*. Order number AD A137 427; December 1983.

Digital Equipment Corporation. *DECnet/OSI Phase V: Making the Transition From Phase IV.* EK-PVTRN-BR; 1989.

———. *DECserver 200 Local Area Transport (LAT) Network Concepts*. AA-LD84A-TK; June 1988.

———. *DIGITAL Network Architecture (Phase V)*. EK-DNAPV-GD-001; September 1987.

Digital Equipment Corporation, Intel Corporation, Xerox Corporation. *The Ethernet, A Local-Area Network, Data Link Layer and Physical Layer Specifications.* Ver. 2.0; November 1982.

Feinler, E.J., et al. *DDN Protocol Handbook*, Vols. 1–4, NIC 50004, 50005, 50006, 50007. Defense Communications Agency. Alexandria, Virginia; December 1985.

Garcia-Luna-Aceves, J.J. "A Unified Approach to Loop-Free Routing Using Distance Vectors or Link States." ACM 089791-332-9/89/0009/0212, pp. 212–223; September 1989.

Hemrick, C. and L. Lang. "Introduction to Switched Multi-megabit Data Service (SMDS), an Early Broadband Service." *Proceedings of the XIII International Switching Symposium* (ISS 90), May 27–June 1, 1990.

Hewlett-Packard Company. X.25: The PSN Connection; An Explanation of Recommendation X.25. 5958-3402; October 1985.

IEEE Project 802—*Local & Metropolitan Area Networks. Proposed Standard: Distributed Queue Dual Bus (DQDB) Subnetwork of a Metropolitan Area Network (MAN)*; February 7, 1990.

IEEE 802.2—*Local Area Networks Standard, 802.2 Logical Link Control*. ANSI/IEEE Standard; October 1985.

IEEE 802.3—*Local Area Networks Standard, 802.3 Carrier Sense Multiple Access.* ANSI/IEEE Standard; October 1985.

International Business Machines Corporation. ACF/NCP/VS network control program, system support programs: general information. GC30-3058.

———. *Advanced Communications Function for VTAM (ACF/VTAM), general information: introduction*. GS27-0462.

———. *Advanced Communications Function for VTAM, general information: concepts. GS27-0463.*

———. *Dictionary of Computing.* SC20-1699-7; 1987.

———. *Local Area Network Technical Reference*. SC30-3883.

———. *Network Problem Determination Application: general information.* GC34-2010.

———. *Synchronous Data Link Control: general information*. GA27-3093.

———. *Systems Network Architecture: concepts and products.* GC30-3072.

———. *Systems Network Architecture: technical overview.* GC30-3073-1; 1985.

———. *Token-Ring Network Architecture Reference.* SC30-3374.

———. *Token-Ring Problem Determination Guide*. SX27-3710-04; 1990.

International Organization for Standardization. *Information Processing System—Open System Interconnection; Specification of Abstract Syntax Notation One (ASN.1)*. International Standard 8824; December 1987.

McGraw-Hill/Data Communications. *McGraw-Hill's Compilation of Data Communications Standards.* Edition III; 1986.

National Security Agency. *Blacker Interface Control Document.* March 21, 1989.

Novell, Inc. IPX Router Specification, Version 1.10. Part Number 107-000029-001. October 16, 1992.

———. NetWare Link Services Protocol (NLSP) Specification, Revision 0.9. Part Number 100-001708-001. March 1993.

StrataCom. *Frame Relay Specification with Extensions.* 001-208966, Rev.1.0; September 18, 1990.

Xerox Corporation. *Internet Transport Protocols.* XNSS 029101; January 1991.

APPENDIX B

Ethernet Type Codes

This appendix lists known Ethernet type codes for use with type code filtering configuration commands.

Note Not all codes are used on both Token Ring or Ethernet media.

Hexadecimal	Description (Notes)
0000-05DC	IEEE 802.3 Length Field
0101-01FF	Experimental; for development (Conflicts with 802.3 length fields)
0200	Xerox PUP (Conflicts with IEEE 802.3 length fields)
0201	Xerox PUP Address Translation (Conflicts with IEEE 802.3 length fields)
0600	Xerox XNS IDP
0800	DoD Internet Protocol (IP) *[1] #[2]
0801	X.75 Internet
0802	NBS Internet
0803	ECMA Internet
0804	CHAOSnet
0805	X.25 Level 3
0806	Address Resolution Protocol (for IP and CHAOS)
0807	XNS Compatibility
081C	Symbolics Private
0888-088A	Xyplex
0900	Ungermann-Bass Network Debugger
0A00	Xerox IEEE 802.3 PUP
0A01	Xerox IEEE 802.3 PUP Address Translation
0BAD	Banyan VINES IP
0BAE	Banyan VINES Loopback
0BAF	Banyan VINES Echo
1000	Berkeley trailer negotiation

Hexadecimal	Description (Notes)
1001-100F	Berkeley trailer encapsulation for IP
1600	VALID system protocol
4242	PCS Basic Block Protocol
5208	BBN Simnet Private
6000	DEC unassigned
6001	DEC Maintenance Operation Protocol (MOP) Dump/Load Assistance
6002	DEC MOP Remote Console
6003	DEC DECnet Phase IV
6004	DEC Local Area Transport (LAT)
6005	DEC DECnet Diagnostics
6006	DEC DECnet Customer Use
6007	DEC Local Area VAX Cluster (LAVC)
6008	DEC unassigned
6009	DEC unassigned
6010-6014	3Com Corporation
7000	Ungermann-Bass (UB) Download
7001	UB diagnostic/loopback
7002	UB diagnostic/loopback
7020-7029	LRT (England)
7030	Proteon
8003	Cronus VLN
8004	Cronus Direct
8005	HP Probe protocol
8006	Nestar
8008	AT&T
8010	Excelan
8013	Silicon Graphics diagnostic (obsolete)
8014	Silicon Graphics network games (obsolete)
8015	Silicon Graphics reserved type (obsolete)
8016	Silicon Graphics XNS NameServer, bounce server (obsolete)
8019	Apollo Domain
802E	Tymshare
802F	Tigan, Inc.
8035	Reverse Address Resolution Protocol (RARP)
8036	Aeonic Systems
8038	DEC LANBridge Management
8039	DEC unassigned
803A	DEC unassigned
803B	DEC unassigned

Hexadecimal	Description (Notes)
803C	DEC unassigned
803D	DEC Ethernet CSMA/CD Encryption Protocol
803E	DEC unassigned
803F	DEC LAN Traffic Monitor Protocol
8040	DEC unassigned
8041	DEC unassigned
8042	DEC unassigned
8044	Planning Research Corporation
8046-8047	AT&T
8049	ExperData (France)
805B	Versatile Message Translation Protocol RFC-1045 (Stanford)
805C	Stanford V Kernel, production
805D	Evans & Sutherland
8060	Little Machines
8062	Counterpoint Computers
8065-8066	University of Mass. at Amherst
8067	Veeco Integrated Automation
8068	General Dynamics
8069	AT&T
806A	Autophon (Switzerland)
806C	ComDesign
806D	Compugraphic Corporation
806E-8077	Landmark Graphics Corporation
807A	Matra (France)
807C	Merit Internodal
807D-8080	Vitalink Communications
8080	Vitalink TransLAN III Management
8081-8083	Counterpoint Computers
8088-808A	Xyplex
809B	Kinetics EtherTalk (AppleTalk over Ethernet)
809C-809E	Datability
809F	Spider Systems Ltd.
80A3	Nixdorf Computers (West Germany)
80A4-80B3	Siemens Gammasonics Inc.
80C0-80C3	Digital Communications Assoc. Inc.
80C1	DCA Data Exchange Cluster
80C4	Banyan VINES IP
80C5	Banyan VINES Echo
80C6	Pacer Software

Hexadecimal	Description (Notes)
80C7	Applitek Corporation
80C8-80CC	Intergraph Corporation
80CD-80CE	Harris Corporation
80CF-80D2	Taylor Instrument
80D3-80D4	Rosemount Corporation
80D5	IBM SNA Services over Ethernet
80DD	Varian Associates
80DE	Integrated Solutions Transparent Remote File System (TRFS)
80DF	Integrated Solutions
80E0-80E3	Allen-Bradley
80E4-80F0	Datability
80F2	Retix
80F3	Kinetics AppleTalk Address Resolution Protocol (AARP)
80F4-80F5	Kinetics
80F7	Apollo Computer
80FF-8103	Wellfleet Communications
8069	AT&T
807B	Dansk Data Elektronik A/S
8107	Symbolics Private
8108	Symbolics Private
8109	Symbolics Private
8130	Waterloo Microsystems Inc.
8131	VG Laboratory Systems
8137	Novell NetWare IPX (old)
8137-8138	Novell, Inc.
8139-813D	KTI
9000	Loopback (Configuration Test Protocol)
9001	Bridge Communications XNS Systems Management
9002	Bridge Communications TCP/IP Systems Management
FF00	BBN VITAL LANBridge cache wakeups

1. An asterisk (*) indicates the current connection in various informational displays.
2. A pound sign (#) is a delimiting character for configuration commands that contain arbitrary text strings.

APPENDIX C

Regular Expressions

This appendix explains what regular expressions are and how you use them in router configurations. It also gives you details for composing regular expressions. This appendix has the following sections:

- General Concepts
- Using Regular Expressions
- Creating Regular Expressions
- Practical Examples

General Concepts

A regular expression is a pattern to match against an input string. You specify the pattern that a string must match when you compose a regular expression. Matching a string to the specified pattern is called “pattern matching.” Pattern matching either succeeds or fails.

For example, you can specify in an X.25 routing table that incoming packets with destination addresses beginning with 3107 get routed to serial interface 0. In this example, the pattern to match is the *3107* specified in the X.25 routing table. The string is the initial portion of the destination address of any incoming X.25 packet. When the destination address (string) matches *3107* (pattern), then pattern matching succeeds, and the IOS software routes the packet to serial interface 0. When the initial portion of the destination address does not match *3107*, then pattern matching fails, and the IOS software does not route the packet to serial interface 0.

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

Using Regular Expressions

There are several implementations of regular expressions in Cisco router configurations. Generally, you use regular expressions in the following ways:

- To specify chat scripts for asynchronous lines in the dial-on-demand routing (DDR) feature
- To specify routes in a routing table for the X.25 switching feature
- To filter packets and routing information in the DECnet and BGP protocols

Specifying Chat Scripts

On asynchronous lines, chat scripts send commands for modem dialing and logging in to remote systems. You use a regular expression in the **modem chat-script** command to specify the name of the chat script that the 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.

For configuration information on chat scripts, refer to the *Router Products Configuration Guide*, the "Configuring DDR" chapter. For details on the **modem chat-script** and **dialer map** commands, refer to the "DDR Commands" chapter of the *Router Products Command Reference* publication.

Specifying Routes in a Routing Table

As described in the "General Concepts" section, you can use regular expressions to help specify routes in an X.25 routing table. When you create entries in an X.25 routing table, you can use regular expressions in the **x25 route** command to help specify routes for incoming calls. When a router receives an incoming call that should be forwarded to its destination, the IOS software consults the X.25 routing table to determine the route. The IOS software compares the X.121 network interface address (or destination address) field and the Call User Data (CUD) field of the incoming packet with the routing table to determine the route. When the destination address and the CUD of the incoming packet match the X.121 and CUD regular expressions you specified in the routing table, the router forwards the call.

For details on creating an X.25 routing table, refer to the *Router Products Configuration Guide*, the "Configuring X.25 and LAPB" chapter. Also see the **x25 route** command in the "X.25 and LAPB Commands" chapter of the *Router Products Command Reference* publication.

Filtering Packets and Routing Information

You can use regular expressions in access lists for both DECnet and BGP. In DECnet, you can use regular expressions in the **access-list** command to filter *connect initiate* packets. With these packets, you can filter packets by DECnet object type, such as MAIL. In BGP, you use regular expressions in the **ip as-path access-list** command for path filtering by neighbor. Using regular expressions, you specify an access list filter on both incoming and outbound updates based on the BGP autonomous system paths.

For configuration information on filtering connect initiate packets and path filtering by neighbor, refer to the *Router Products Configuration Guide*, the "Configuring DECnet" and "Configuring IP Routing Protocols" chapters.

For detailed information on the **access-list** and **ip as-path access-list** commands, refer to the "DECnet Commands" and "IP Routing Protocols Commands" chapters of the *Router Products Command Reference* publication.

Creating Regular Expressions

A regular expression can be a single-character pattern or a multiple-character pattern. That is, a regular expression can be a single character that matches the same single character in the input string or multiple characters that match the same multiple characters in the input string. This section describes creating both single-character patterns and multiple-character patterns. It also discusses creating more complex regular expressions using multipliers, alternation, anchoring, and parentheses.

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-Z, a-z) or number (0-9) as a single-character pattern. The following examples are single-character regular expression patterns:

A

k

5

You can also use other keyboard characters (such as ! or ~) as single-character patterns, but certain keyboard characters have special meaning when used in regular expressions. Table C-1 lists the keyboard characters with special meaning.

Table C-1 Characters with Special Meaning

Character	Special Meaning
.	Matches any single character, including white space.
*	Matches 0 or more sequences of the pattern.
+	Matches 1 or more sequences of the pattern.
?	Matches 0 or 1 occurrences of the pattern.
^	Matches the beginning of the input string.
$	Matches the end of the input string.
_ (underscore)	Matches a comma (,), left brace ({), right brace (}), left parenthesis, right parenthesis, the beginning of the input string, the end of the input string, or a space.

To use these special characters 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 lower- or uppercase alphabet.

You can simplify ranges by typing only the end points of the range separated by a dash (-). Simplify the previous range as follows:

[a-dA-D]

To add a dash as a single-character pattern in your range, include another dash 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 lower- or uppercase alphabet, a dash, or a right square bracket.

You can reverse the matching of the range by including a caret (^) 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 dot key to match the letter *a* followed by any single character. With this example, the strings *ab*, *a!*, or *a2* are all valid matches for the regular expression.

You can remove the special meaning of the dot character by preceding it with a backslash. In the expression **a\.** only the string *a.* matches the regular expression.

You can create a multiple-character regular expressions containing all letters, all digits, all keyboard characters, or a combination of letters, digits, and other keyboard characters. The following examples are all valid regular expressions:

telebit

3107

v32bis

Multipliers

You can create more complex regular expressions that instruct the 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. Table C-2 lists the special characters that specify “multiples” of a regular expression.

Table C-2 Special Characters Used as Multipliers

Character	Description
*	Matches 0 or more single- or multiple-character patterns.
+	Matches 1 or more single- or multiple-character patterns.
?	Matches 0 or 1 occurrences of the single- or multiple-character pattern.

The following example matches any number of occurrences of the letter *a*, including none:

a*

The following pattern requires there to be at least one letter *a* in the string to be matched:

a+

The following pattern matches the string *bb* or *bab*:

ba?b

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])+

The order for matches using multipliers (*, +, or ?) is longest construct first. Nested constructs are matched from outside to inside. Concatenated constructs are matched beginning at the left side of the construct. Thus, the regular expression matches *A9b3*, but not *9Ab3* because the alphabet is given first in the construct.

Alternation

Alternation allows you to specify alternative patterns to match against a string. You 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 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. You “anchor” these regular expressions to a portion of the input string using the special characters shown in Table C-3.

Table C-3 Special Characters Used for Anchoring

Character	Description
^	Matches the beginning of the input string.
$	Matches the end of the input string.

Note another use for the ^ symbol. 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 ($), parentheses (( )), 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.

Using the underscore character, you can replace long regular expression lists. For example, you can replace the following list of regular expressions with simply **_1300_**:

^1300$

^1300(space)

(space)1300

{1300,

,1300,

{1300}

,1300,

(1300

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 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 a 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. So, the regular expression can match *aZbcTZT*. The software remembers that character #1 is *Z* and character #2 is *T* and then uses *Z* and *T* again later in the regular expression.

The parentheses do not change whether the pattern matches the input string or not, they only instruct the software to recall that part of the matched string. So, the regular expression **(a)b** still matches the input string *ab*, and **(^3107)** still matches a string beginning with *3107*, but now the IOS software can recall the *a* of the *ab* string and the starting*3107* of another string for use later.

Practical Examples

This section shows you practical examples of regular expressions. The examples correspond with the various ways you can use regular expressions in your configurations.

The following example uses regular expressions in the **modem chat-script** command to specify chat scripts for lines connected to Telebit and U.S. Robotics modems. The regular expressions are **telebit.*** and **usr.***. When the chat script name (the string) matches the regular expression (the pattern specified in the command), then the IOS software uses that chat script for the specified lines. For lines 1 and 6, the IOS software uses the chat script named *telebit* followed by any number of occurrences (*) of any character (.). For lines 7 and 12, the IOS software uses the chat script named *usr* followed by any number of occurrences (*) of any character (.).

```
! Some lines have Telebit modems
line 1 6
modem chat-script telebit.*
! Some lines have US Robotics modems
line 7 12
modem chat-script usr.*
```

In the following X.25 switching feature 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 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
```

In the following DECnet example, the regular expression is ^^**SYSTEM$**. The access list permits access to all connect initiate packets that match the access identification of SYSTEM.

```
access-list 300 permit 0.0 63.1023 eq id ^SYSTEM$
```

The following BGP example contains the regular expression **^123.***. The example specifies that BGP neighbor with IP address 128.125.1.1 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 131.108.0.0
neighbor 129.140.6.6 remote-as 123
neighbor 128.125.1.1 remote-as 47
neighbor 18.125.1.1 filter-list 1 out
```

APPENDIX D

ASCII Character Set

Some commands described in this manual require that you enter the decimal representation of an ASCII character. Table D-1 provides code translations from ASCII characters to decimal numbers. For example, the ASCII carriage return (CR) is decimal 13. This means that pressing Ctrl-M at your terminal generates decimal 13, which is interpreted as a CR.

Table D-1 ASCII Translation Table

Numeric Values		ASCII		Keyboard
Decimal	Hex	Character	Meaning	Entry
0	00	NUL	Null	Ctrl-@
1	01	SOH	Start of heading	Ctrl-A
2	02	STX	Start of text	Ctrl-B
3	03	ETX	Break/end of text	Ctrl-C
4	04	EOT	End of transmission	Ctrl-D
5	05	ENQ	Enquiry	Ctrl-E
6	06	ACK	Positive acknowledgment	Ctrl-F
7	07	BEL	Bell	Ctrl-G
8	08	BS	Backspace	Ctrl-H
9	09	HT	Horizontal tab	Ctrl-I
10	0A	LF	Line feed	Ctrl-J
11	0B	VT	Vertical tab	Ctrl-K
12	0C	FF	Form feed	Ctrl-L
13	0D	CR	Carriage return	Ctrl-M
14	0E	SO	Shift out	Ctrl-N
15	0F	SI	Shift in/XON (resume output)	Ctrl-O
16	10	DLE	Data link escape	Ctrl-P
17	11	DC1	Device control character 1	Ctrl-Q
18	12	DC2	Device control character 2	Ctrl-R
19	13	DC3	Device control character 3	Ctrl-S
20	14	DC4	Device control character 4	Ctrl-T
21	15	NAK	Negative Acknowledgment	Ctrl-U

Numeric Values		ASCII		Keyboard
Decimal	Hex	Character	Meaning	Entry
22	16	SYN	Synchronous idle	Ctrl-V
23	17	ETB	End of transmission block	Ctrl-W
24	18	CAN	Cancel	Ctrl-X
25	19	EM	End of medium	Ctrl-Y
26	1A	SUB	substitute/end of file	Ctrl-Z
27	1B	ESC	Escape	Ctrl-[
28	1C	FS	File separator	Ctrl-\
29	1D	GS	Group separator	Ctrl-]
30	1E	RS	Record separator	Ctrl-^
31	1F	US	Unit separator	Ctrl-_
32	20	SP	Space	Space
33	21	!	!	!
34	22	"	"	"
35	23	#	#	#
36	24	$	$	$
37	25	%	%	%
38	26	&	&	&
39	27	’	’	’
40	28	(	(	(
41	29	)	)	)
42	2A	*	*	*
43	2B	+	+	+
44	2C	,	,	,
45	2D	-	-	-
46	2E	.	.	.
47	2F	/	/	/
48	30	0	Zero	0
49	31	1	One	1
50	32	2	Two	2
51	33	3	Three	3
52	34	4	Four	4
53	35	5	Five	5
54	36	6	Six	6
55	37	7	Seven	7
56	38	8	Eight	8
57	39	9	Nine	9
58	3A	:	:	:
59	3B	;	;	;

Numeric Values		ASCII		Keyboard
Decimal	Hex	Character	Meaning	Entry
60	3C	<	<	<
61	3D	=	=	=
62	3E	>	>	>
63	3F	?	?	?
64	40	@	@	@
65	41	A	A	A
66	42	B	B	B
67	43	C	C	C
68	44	D	D	D
69	45	E	E	E
70	46	F	F	F
71	47	G	G	G
72	48	H	H	H
73	49	I	I	I
74	4A	J	J	J
75	4B	K	K	K
76	4C	L	L	L
77	4D	M	M	M
78	4E	N	N	N
79	4F	O	O	O
80	50	P	P	P
81	51	Q	Q	Q
82	52	R	R	R
83	53	S	S	S
84	54	T	T	T
85	55	U	U	U
86	56	V	V	V
87	57	W	W	W
88	58	X	X	X
89	59	Y	Y	Y
90	5A	Z	Z	Z
91	5B	[	[	[
92	5C	\	\	\
93	5D	]	]	]
94	5E	^	^	^
95	5F	_	_	_
96	60	`	`	`
97	61	a	a	a

Numeric Values		ASCII		Keyboard
Decimal	Hex	Character	Meaning	Entry
98	62	b	b	b
99	63	c	c	c
100	64	d	d	d
101	65	e	e	e
102	66	f	f	f
103	67	g	g	g
104	68	h	h	h
105	69	i	i	i
106	6A	j	j	j
107	6B	k	k	k
108	6C	l	l	l
109	6D	m	m	m
110	6E	n	n	n
111	6F	o	o	o
112	70	p	p	p
113	71	q	q	q
114	72	r	r	r
115	73	s	s	s
116	74	t	t	t
117	75	u	u	u
118	76	v	v	v
119	77	w	w	w
120	78	x	x	x
121	79	y	y	y
122	7A	z	z	z
123	7B	{	{	{
124	7C	|	|	|
125	7D	}	}	}
126	7E	~	Tilde	~
127	7F	DEL	Delete	Del

APPENDIX E

X.3 PAD Parameters

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

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

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

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

This appendix discusses the X.3 PAD parameters. The *Cisco Access Connection Guide* explains how to make PAD connections and how to switch between connections.

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

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

Some of the commands described in this section require ASCII decimal values, which are listed in the appendix "ASCII Character Set," earlier in this publication.

Parameter 1: Escape from Data Transfer (Not Supported)

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

Parameter 2: Local Echo Mode

Parameter 2 determines whether or not PAD is required to perform local echo of characters. This parameter can be negotiated end-to-end on translation sessions. On incoming PAD connections, software turns local echo off on the remote PAD to support the Cisco user interface. See Table E-1 for local echo mode values and their descriptions.

Table E-1 PAD Local Echo Mode Values

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

Parameter 3: Data Forward Character

Parameter 3 sets up a packet forwarding mask; that is, it selects which character causes PAD to forward a packet either before expiration of the Idle Timer (see parameter 4) or when in local editing mode. See Table E-2 for data forward character values and their descriptions.

Table E-2 PAD Data Forward Character Values

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

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

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

Parameter 4: Idle Timer

Parameter 4 sets the length of time the software waits for new data before sending a packet in the absence of a data forwarding character. See Table E-3 for idle timer values and their descriptions.

Table E-3 PAD Idle Timer Values

Value	Description
0	No timer.
1-255	Delay value in twentieths of a second (default for both connection types is 1).

Parameter 5: Device Control (Not Supported)

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

Parameter 6: PAD Service Signals (Not Supported)

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

Parameter 7: Action upon Receipt of a BREAK Signal

Parameter 7 defines the action of the PAD after receiving a BREAK signal from the terminal. See Table E-4 for PAD BREAK signal values and their descriptions.

Table E-4 PAD BREAK Signal Values

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

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

Parameter 8: Discard Output

Parameter 8 indicates to the PAD whether to discard received packets rather than disassemble and transmit them. This parameter works in conjunction with parameter 7. If value 16 is chosen for parameter 7, all output is discarded after reception of the BREAK signal. Setting parameter 8 to zero restores normal data delivery to the terminal. See Table E-5 for PAD discard output values and their descriptions.

Table E-5 PAD Discard Output Values

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

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

Parameter 9: Return Padding (Not Supported)

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

Parameter 10: Line Folding (Not Supported)

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

Parameter 11: Baud Rate

Parameter 11 is a read-only value that determines the baud rate transmitted across the interface between PAD and the terminal. See Table E-6 for PAD baud rate values and their descriptions.

Table E-6 PAD Baud Rate Values

Value	Description (in bits per second)
10	50
5	75
9	100
0	110
1	134.5
6	150
8	200
2	300
4	600
3	1200
7	1800

Value	Description (in bits per second)
11	75/1200
12	2400
13	4800
14	9600
15	19200
16	48000
17	56000
18	64000

Parameter 12: Input Flow Control (Not Supported)

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

Parameter 13: LINE FEED Insertion

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

Table E-7 PAD LINE FEED Signal Values

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

Parameter 14: LINE FEED Padding (Not Supported)

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

Parameter 15: Local Editing

Parameter 15 enables or disables a PAD editing function for the terminal in data transfer mode. Enabling the editing function disables the Idle Timer (see parameter 4). The user at the terminal can make corrections and display the line buffer containing the characters to be transmitted when the forwarding character (see parameter 3) is received. See Table E-8 for PAD local editing function values and their descriptions.

Table E-8 PAD Local Editing Function

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

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

Parameter 16: Character Delete

Parameter 16 allows you to select a character that will delete characters while in PAD editing mode. This character is valid only if parameter 15 is set to one. See Table E-9 for PAD line display editing function values and their descriptions.

Table E-9 PAD Line Display Editing Function

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

Parameter 17: Line Delete

Parameter 17 allows you to select a character that will delete a line while in PAD editing mode. This character is valid only if parameter 15 is set to one. See Table E-10 for PAD line delete editing function values and their descriptions.

Table E-10 PAD Line Delete Editing Function

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

Parameter 18: Line Display

Parameter 18 allows you to select a character that will display a line while in PAD editing mode. This character is valid only if parameter 15 is set to one. See Table E-11 for PAD line display editing function values and their descriptions.

Table E-11 PAD Line Display Editing Function

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

Index

INDEX

Symbols

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

X