
Configuring Protocol Translation

This chapter describes how to configure protocol translation connections using the Cisco IOS software. The protocol translation facility assumes that you understand how to use the configuration software. This chapter 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 X.25, Telnet, Local Area Transport (LAT), TN3270, AppleTalk Remote Access (ARA), Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP), and the XRemote configuration chapters in this publication.

For a complete explanation of the **translate** command, refer to the *Access Services Command Reference*. For more information about making connections and establishing translation sessions, see the “Making Connections to Network Devices” chapter.

In the context of this chapter, a router or access server set up to run protocol translation software is referred to as a *router*.

Note Telnet is a remote terminal protocol that is part of the Transmission Control Protocol/Internet Protocol (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

A router set up for 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. Routers running protocol translation also support X.25 as a transport mechanism for IP, IPX, and ARA packets, and X.3- and X.29-based packet/assembler/disassembler (PAD) connections. This X.25 connection allows transport of TCP/IP packets across the X.25 packet-switching network in the same way that a router running protocol translation would transport them.

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

Routers that do not run protocol translation do not support an X.25 PAD function, so they cannot communicate with hosts directly connected to the X.25 public data network (PDN). The Cisco IOS software encapsulates TCP/IP packets in X.25 packets for transfer over a packet-switching network,

and these packets can be received by routers. Only devices using protocol translation include PAD capabilities, but other devices can communicate with routers running protocol translation using the TCP/IP or LAT protocols. Routers supporting protocol translation can translate TCP/IP and LAT into X.25, and then communicate with X.25 hosts. Routers running protocol translation support all PAD standards (X.28, X.29, and X.3). The connection to the packet-switching network is through a synchronous serial interface.

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 data terminal equipment (DTE) device. By default, all PAD commands and related connections are enabled using the **service pad** global configuration command.

How Protocol Translation Works

The protocol translation software attempts to provide transparent protocol translation between systems running different protocols. It enables terminal users on one network to access hosts on another network, despite differences in the native protocol stacks associated with the originating device and the target host.

A router supports virtual terminal connections in both directions between the protocols in the following list. You can configure the Cisco IOS software to translate automatically between them. This is called the one-step translation method.

- X.25 and Local Area Transport (LAT)
- X.25 and Telnet sessions using the Transmission Control Protocol (TCP)
- LAT and TCP/Telnet

On outgoing connections, you can also use the one-step protocol translation facility to tunnel SLIP or PPP to IP and IPX networks or ARA to AppleTalk networks across X.25, LAT, or IP (on outgoing connections only).

The Cisco IOS software supports limited connections in both directions between the following protocols. Connecting between these protocols requires that you first connect to a router that supports protocol translation and then to the host to which you want to connect. This is called the two-step translation method.

- XRemote to X.25 PAD environments (XRemote must use the two-step method)
- TN3270 to LAT, X.25, and TCP/Telnet (TN3270 must use the two-step method)

The following sections describe the process of tunneling SLIP and PPP using protocol translation, as well as the two-step and the one-step translation methods. See the end of this chapter for protocol translation application and session examples.

One-Step Protocol Translation

Use the one-step method when network users repeatedly log on to the same remote network hosts through a router. This connection is more efficient and enables the device to have more knowledge of the protocols in use, because the router acts as a network connection rather than as a terminal. The one-step method provides transparent protocol conversion. When connecting to the remote network host, the user enters the connection command to the remote network host, but does not need to specify protocol translation. The network administrator has already created a configuration that defines a connection and the protocols to be translated. The user performs only one step to connect with the host.

When you make a one-step connection to the router, the Cisco IOS software determines which host the connection is for and which protocol that host is using. It then establishes a new network connection using the protocol required by that host.

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

To configure the one-step method of protocol translation, set up the following protocols and connection options in the configuration file:

- The incoming connection—The configuration includes the protocol to be used—LAT, X.25, or TCP/IP (Telnet)—the address, and any options such as reverse charging or binary mode that are supported for the incoming connection.
- The outgoing connection—The outgoing connection is defined in the same way as the incoming connection, except that SLIP, PPP (including IP and IPX on PPP sessions), and ARA are also supported.
- The connection features global options—You can specify additional features for the connection to allow, for example, incoming call addresses to match access-list conditions or limit the number of users that can make the connection.

Two-Step Protocol Translation

Use the two-step process to run protocol translation for one-time connections or when you use router as a general-purpose gateway between two types of networks (for example, X.25 PDN and TCP/IP). You must first configure the software for the transmission protocols that you will be using.

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

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

Tunneling of SLIP, PPP, and ARA

Unlike other protocols, such as LAT, X.25, and TCP, which are actually translated when you use one-step protocol translation, SLIP, PPP, and ARA are not translated to the destination protocol. Instead, they are carried inside a LAT, X.25, or TCP tunnel specific to the device on the remote network. However, you use the protocol translation facility to enable tunneling of SLIP, PPP, or ARA.

You can also tunnel IPX-PPP over X.25, TCP, or LAT, to an IPX network when tunneling PPP on virtual terminal (VTY) lines.

One-Step Tunneling of SLIP, PPP, and ARA

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

Refer to the section “One-Step Protocol Translation” in this chapter for more information about one-step protocol translation.

If you are tunneling PPP, SLIP, or ARA across X.25, you must also set up your X.3 profile correctly using the **x29 profile** command, as described in the section “Configure One-Step Tunneling of SLIP or PPP” later in this chapter. For more information about using the **x29 profile** command, refer to the chapter “Protocol Translation Configuration Commands” in the *Access Services Command Reference*.

Two-Step Tunneling of SLIP and PPP

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

To issue SLIP or PPP EXEC commands on VTY lines, refer to the section “Enable SLIP and PPP on Virtual Asynchronous Interfaces” in the chapter “Configuring SLIP and PPP” in this publication.

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

Two-Step Tunneling of ARA

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

For more information about enabling two-step tunneling of ARA on VTY lines, refer to the section “Connect to an AppleTalk Network from a Client Running a Different Virtual Terminal Protocol” in the chapter “Configuring an AppleTalk Remote Access Server” in this publication.

Configure Protocol Translation

Specifically, this section describes how to perform the following tasks:

- Configure Two-Step Protocol Translation
- Configure One-Step Protocol Translation

Configure Two-Step Protocol Translation

To translate using the two-step method, perform the following task beginning in global configuration mode. The first step is required only if you are tunneling SLIP or PPP using the two-step protocol translation facility:

Task	Command
Step 1 Establish an incoming connection to the router running protocol translation.	connect lat pad telnet tunnel ¹
Step 2 Establish the outgoing connection from the router supporting protocol translation to another network host.	connect lat pad telnet tunnel ppp slip ¹

1. Each of these commands is described in the “Connections Commands” chapter of the *Access Services Command Reference*.

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

Configure One-Step Protocol Translation

To create one-step protocol translation connection specifications, perform the following task in global configuration mode:

Task	Command
Create the connection specifications for one-step protocol translation.	translate <i>protocol incoming-address</i> [<i>in-options</i>] <i>protocol outgoing-address</i> [<i>out-options</i>] [<i>global-options</i>]

For incoming PAD connections, the router uses a default PAD profile to set the remote X.3 PAD parameters unless a profile script is defined in the **translate** command. To override the default PAD profile the router uses, you must create a PAD profile scrip using the **x29 profile** global configuration command. In the following example, *default* is the name of the default PAD profile script and *parameter:value* is the X.3 PAD parameter number and value separated by a colon.

```
x29 profile default parameter:value [parameter:value]
```

Change the Number of Supported Translation Sessions

Because each protocol translation session uses a virtual terminal (VTY) line, you need to increase the number of VTY lines to increase the number of protocol translation sessions. That is, if your router has ten VTY lines, you can have up to ten protocol translation sessions. The default number of VTY lines is 5 (lines 0 through 4). To increase the number of lines, and thus the maximum number of protocol translation sessions, perform the following tasks, as appropriate, beginning in global configuration mode:

Task	Command
Increase the number of virtual terminal lines, and thus, the maximum number of protocol translation sessions.	line vty <i>number</i> ¹
Decrease the number of virtual terminal lines, and thus, the maximum number of protocol translation sessions.	no line vty <i>number</i>

1. This command is documented in the “Terminal Lines and Modem Support Commands” chapter in the *Access Services Command Reference*.

Protocol translation is a CPU-intensive task. Increasing the number of protocol translation sessions while routing is enabled can impact available memory. The amount of memory available depends on the platform type, the amount of DRAM available, the activity of each translation session, and the speed of the link. If you are using the maximum number of sessions and have problems with memory, you might need to decrease the number of protocol translation sessions.

The maximum number of protocol translation sessions for each platform can be increased to the number specified in Table 7.

Table 7 Maximum Number of Protocol Translation Sessions by Platform

Platform	Default Number of VTY Lines	MAXLINES ¹	Maximum VTY Lines with PT Option
Cisco 1000 running Cisco IOS software	5	6	5
Cisco 2500 (8 asynchronous ports)	5	200	190
Cisco 2500 (16 asynchronous ports)	5	200	182
Cisco 3000	5	200	198
Cisco 4000	5	200	198
Cisco 4500	5	1002	1000
Cisco 4700	5	1002	1000
Cisco 7000	5	120	118
Cisco 7000 with RSP	5	1002	1000
irix	5	7	N/A
UNIX Sun and Solaris OS	5	7	N/A

1. MAXLINES = Maximum number of VTYs + (TTYs + AUX + CON lines).

Configure Tunneling of SLIP, PPP, or ARA

This section describes how to perform the following tasks:

- Configure Two-Step Tunneling of SLIP or PPP
- Configure One-Step Tunneling of SLIP or PPP
- Configure One-Step Tunneling of ARA

You can also enable IPX over tunneled PPP sessions.

Configure Two-Step Tunneling of SLIP or PPP

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

Task	Command
Step 1 Enable tunneling of SLIP and PPP using two-step protocol translation.	vty-async
Step 2 Exit from global configuration mode into EXEC mode.	exit
Step 3 Establish an incoming connection to the router running protocol translation.	connect lat pad telnet tunnel
Step 4 Establish the outgoing connection from the router supporting protocol translation to another network host.	connect slip ppp tunnel

If you want to configure IPX over your PPP sessions on VTY lines, refer to the section “Enable IPX over PPP on Virtual Asynchronous Interfaces” in the chapter “Configuring SLIP or PPP.”

To configure two-step tunneling of ARA on VTY lines, refer to the section “Connect to an AppleTalk Network from a Client Running a Different Virtual Terminal Protocol” in the chapter “Configuring an AppleTalk Remote Access Server” in this publication.

Configure One-Step Tunneling of SLIP or PPP

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

Task	Command
(Optional.) If you tunneling PPP over X.25, create an X.3 Profile so that the router will interoperate with the PAD.	x29 profile name parameter:value [parameter:value]¹
Create the connection specifications for one-step protocol translation.	translate protocol incoming-address [in-options] protocol outgoing-address [out-options] [global-options]

1. This command is documented in the “X.25 Configuration Commands” chapter of the *Wide-Area Networking Command Reference*.

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

1:0, 2:0, 3:2, 4:1, 5:0, 6:0, 7:21, 8:0, 9:0, 10:0, 11:14, 12:0, 13:0, 14:0, 15:0, 16:127, 17:24, 18:18, 19:0, 20:0, 21:0, 22:0

For more information about creating an X.29 profile script, see the section “Create an X.29 Profile Script” in this chapter. For an example of configuring PPP over X.25, refer to the section “Tunneling PPP Over X.25 Example.”

You can configure an outgoing session for IPX–PPP. To do so, issue the **ipx loopback number** option for the outgoing session. For information about this option, refer to the chapter “Protocol Translation Configuration Commands” in the *Access Services Command Reference*.

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

SLIP and PPP, including IPX–PPP, can be tunnelled on outgoing connections only.

Enable Dynamic Address Assignment for Outgoing SLIP and PPP on VTY Lines

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

Assign IP Addresses Using DHCP

The Dynamic Host Control Protocol (DHCP) client-proxy feature manages a pool of IP addresses available to PPP or SLIP dial-in clients without a known IP address. This allows a finite number of IP addresses to be reused quickly and efficiently by many clients. Additional benefits include the ability to maintain sessions, such as Telnet, even when a modem line fails. When the client is auto-dialed back into the access server or router, the session can be resumed because the same IP address is reissued to the client by the access server or router.

A DHCP proxy client is a Cisco access server or router configured to arbitrate DHCP calls between a DHCP server and a DHCP client. For more information about DHCP proxy clients, refer to the section “Configure DHCP” in the “Configuring Interfaces” chapter in the *Configuration Fundamentals Configuration Guide*.

Task	Command
Step 1 Specify that the router use the DHCP client-proxy.	ip address-pool dhcp-proxy-client
Step 2 Specify DHCP pooling for the SLIP or PPP client on the outgoing session.	translate protocol incoming-address [in-options] {slip ppp} ip-pool

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

Assign IP Addresses Using Local IP Address Pooling

You can make temporary IP addresses available for outgoing SLIP and PPP clients on outgoing sessions. To do so, you must first specify that the Cisco IOS software use a local IP address pool on all asynchronous interfaces and create one or more local IP address pools (refer to the section “Configure a Local IP Address Pool” in the chapter “Configuring Interfaces” in the *Configuration*

Fundamentals Configuration Guide). You then assign local pooling as part of the **translate** command. To assign IP addresses dynamically on a virtual asynchronous connection, perform the following task, beginning in global configuration mode:

Task	Command
Step 1 Specify that the router use a local IP address pool on all asynchronous interfaces.	ip address-pool local
Step 2 Create one or more local IP address pools.	ip local-pool name begin-ip-address-range [end-ip-address-range]
Step 3 Specify local pooling for the SLIP or PPP client on the outgoing session.	translate protocol incoming-address [in-options] slip ppp ip-pool [scope-name name]

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

Configure One-Step Tunneling of ARA

To tunnel ARA using the one-step protocol translation facility, perform the following tasks, beginning in global configuration mode. The first four steps are required. The next seven steps (5 through 11) are optional:

Task	Command
Step 1 Turn on AppleTalk routing.	appletalk routing
Step 2 Use the protocol translation facility to enable an ARA tunnel across a remote network.	translate protocol incoming-address [in-options] autocommand arap
Step 3 Enter line configuration mode.	line vty line-number [ending-line-number]
Step 4 Enable ARA on one or more lines.	arap enable
Step 5 Set one or more dedicated ARA lines.	arap dedicated
Step 6 Set the session time limit.	arap timelimit [minutes]
Step 7 Set the disconnect warning time.	arap warningtime [minutes]
Step 8 Disallow guests.	arap noguest
Step 9 Require manual password entry.	arap require-manual-password
Step 10 Limit the zones the Macintosh user sees.	arap zonelist zone-access-list-number
Step 11 Control access to networks.	arap net-access-list net-access-list number

Create X.29 Access Lists

The Cisco IOS software provides access lists, to limit access to a router from certain X.25 hosts. Access lists take advantage of the message field defined by “Recommendation X.29,” which describes procedures for exchanging data between two PADs or between a PAD and a DTE device.

To define X.29 access lists, perform the following tasks:

- 1 Create an access list.
- 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 router) and the addresses in an access list.	x29 access-list <i>access-list-number</i> { permit deny } <i>regular-expression</i>

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 router) and the addresses in an access list.	access-class <i>number in</i> ¹

1. This command is documented in the “LAT Configuration Commands” chapter in the *Access Services Command Reference*.

The access-list number is used for incoming TCP access and incoming PAD access. For TCP access, the access server or router using protocol translation uses the defined IP access lists. For incoming PAD connections, the same X.29 access list is used. If you want to 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 “Tunneling PPP Over X.25 Example” at the end of this chapter.

Create an X.29 Profile Script

You can create an X.29 profile script for the **translate** command to use. An X.29 profile script uses X.3 PAD parameters. When an X.25 connection is established, the Cisco IOS software with protocol translation functions as if an X.29 SET PARAMETER packet, which contains 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 <i>parameter:value</i> [<i>parameter:value</i>]

For incoming PAD connections, the router running protocol translation uses a default PAD profile to set the remote X.3 PAD parameters, unless a profile script is defined in the **translate** command. To override the default PAD profile the router uses, you must create a PAD profile script named *default* by using the **x29 profile default** *parameter:value* [*parameter:value*] global configuration command, where *default* is the name of the default PAD profile script and *parameter:value* is the X.3 PAD parameter number and value separated by a colon. For more information about X.3 PAD parameters, refer to the appendix “X.3 PAD Parameters” in the *Access Services Command Reference*.

You can also create an X.29 profile script when connecting to a PAD using the **pad** [/profile name] EXEC command, which is described in the “Connection Commands” chapter of the *Access Services Command Reference*.

Define X.25 Host Names

This section describes how to define symbolic host names. This means that instead of remembering a long numeric address for an X.25 host, you can refer to the X.25 host using a symbolic host name. To define a symbolic host name, perform the following task in global configuration mode:

Task	Command
Define a symbolic host name.	x25 host name <i>x.121-address</i> [cud <i>call-user-data</i>]

Monitoring Protocol Translation Connections

This section describes how to log significant VTY-asynchronous authentication information, such as the X.121 calling address, Call User Data (CUD), and the IP address assigned to a VTY asynchronous connection. Depending on how you configure the logging information to be displayed, you can direct this authentication information to the console, an internal buffer, or a UNIX syslog server. This authentication information can be used to associate an incoming PAD VTY-asynchronous connection with an IP address.

Note By default, the Cisco IOS software displays all messages to the console terminal.

This section describes how to perform the following tasks:

- Log VTY-Async Authentication Information to the Console Terminal
- Log VTY-Async Authentication Information to a Buffer
- Log VTY-Async Authentication Information to a UNIX Syslog Server

Log VTY-Async Authentication Information to the Console Terminal

To log significant VTY-asynchronous authentication information to the console terminal, perform the following task in global configuration mode:

Task	Command
Log significant VTY-asynchronous authentication information.	service pt-vty-logging

Log VTY-Async Authentication Information to a Buffer

To log significant VTY-asynchronous authentication information to a buffer, perform the following task in global configuration mode:

Task	Command
Log significant VTY-asynchronous authentication information.	service pt-vty-logging
Direct the authentication log information to a buffer.	logging buffered <i>[size]</i>

Log VTY-Async Authentication Information to a UNIX Syslog Server

To log significant VTY-asynchronous authentication information to a UNIX syslog server, perform the following task in global configuration mode:

Task	Command
Log significant VTY-asynchronous authentication information.	service pt-vty-logging
Direct the authentication log information to a UNIX syslog server.	logging host

Protocol Translation Session Examples

This section illustrates how to make connections for protocol translation using the one-step and two-step methods.

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

This example illustrates one-step protocol translation featuring a UNIX workstation user making a connection to a remote X.25 host named *host1* over an X.25 PDN. The router automatically converts the Telnet connection request to an X.25 connection request and transmits the request as specified in the system configuration.

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

```
unix% telnet host1
```

Note This example implicitly assumes that the name *host1* is known to the UNIX host (obtained via DNS, IEN116, or a static table) and is mapped to the IP address used in a **translate** command.

The router accepts the Telnet connection and immediately forms an outgoing connection with remote *host1* as defined in a **translate** command.

Next, *host1* sets several X.3 parameters, including local echo. Since the Telnet connection is already set to local echo (at the UNIX host), no changes are made on the TCP connection.

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

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

The *host1* host immediately closes the X.25 connection. The Cisco IOS software then drops the TCP connection, leaving the user back at the UNIX system prompt. Using the Two-Step Method for TCP-to-PAD Connections.

To use the two-step method, perform the following steps:

Step 1 Connect directly from a terminal or workstation to a router.

For example, you might make the following connection requests at a UNIX workstation as a first step to logging into a database called *Information Place* on an X.25 PDN:

```
unix% telnet orion
```

If the router named *orion* is accessible, it returns a login message and you enter your login name and password.

Step 2 Connect from the router to *Information Place*, which is on an X.25 host. You connect to an X.25 host using the **pad** EXEC command followed by the service address:

```
orion> pad 71330
```

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

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

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

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

Changing Parameters and Settings Dynamically

The following example illustrates how to make a dynamic change during a protocol translation session. In this example, you need to edit information on remote host *Information Place*. Suppose that you need to change the X.3 PAD parameters that define the editing characters from the default Delete key setting to the Ctrl-D sequence.

Step 1 Enter the escape sequence to return to the system EXEC prompt:

```
Ctrl ^ x
```

Step 2 Enter the **resume** command with the **/set** keyword and the desired X.3 parameters. X.3 parameter 16 sets the Delete function. ASCII character 4 is the Ctrl-D sequence.

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

The session resumes with the new settings, but now you notice that information is not being displayed correctly. You might want to set the **/debug** switch to check that your parameter setting has not been changed by the host PAD.

Step 3 Enter the escape sequence to return to the system EXEC prompt, then enter the resume command with the **/debug** switch.

```
router > resume /debug
```

The **/debug** switch provides helpful information about the connection.

You can also set a packet dispatch character or sequence using the **terminal dispatch-character** command.

The following example shows how to set ESC (ASCII character 27) as a dispatch character:

```
router > terminal dispatch-character 27
```

To return to the PAD connection, enter the following:

```
router > resume
```

Protocol Translation Application Examples

This section provides protocol translation examples for the following applications:

- Assign Addresses Dynamically for PPP Example
- Basic Configuration Example
- Central Site Protocol Translation Example
- Decreasing the Number of Translation Sessions Example
- Increasing the Number of Translation Sessions Example
- LAT-to-LAT over an IP WAN Example
- LAT-to-LAT over Frame Relay or SMDS Example
- LAT-to-LAT Translation over a WAN Example
- LAT-to-LAT over an X.25 Translation Example
- LAT-to-TCP Translation over a WAN Example
- LAT-to-TCP over an X.25 Example
- LAT-to-X.25 Host Example
- Local IP Address Pool Example

- Local LAT-to-TCP Translation Example
- Local LAT-to-TCP Example
- Standalone LAT-to-TCP Translation Example
- Tunneling SLIP inside TCP Example
- Tunneling PPP Over X.25 Example
- X.25 PAD-to-LAT Example
- X.25 PAD-to-TCP Example
- X.29 Access List Example
- X.3 Profile Example

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

Assign Addresses Dynamically for PPP Example

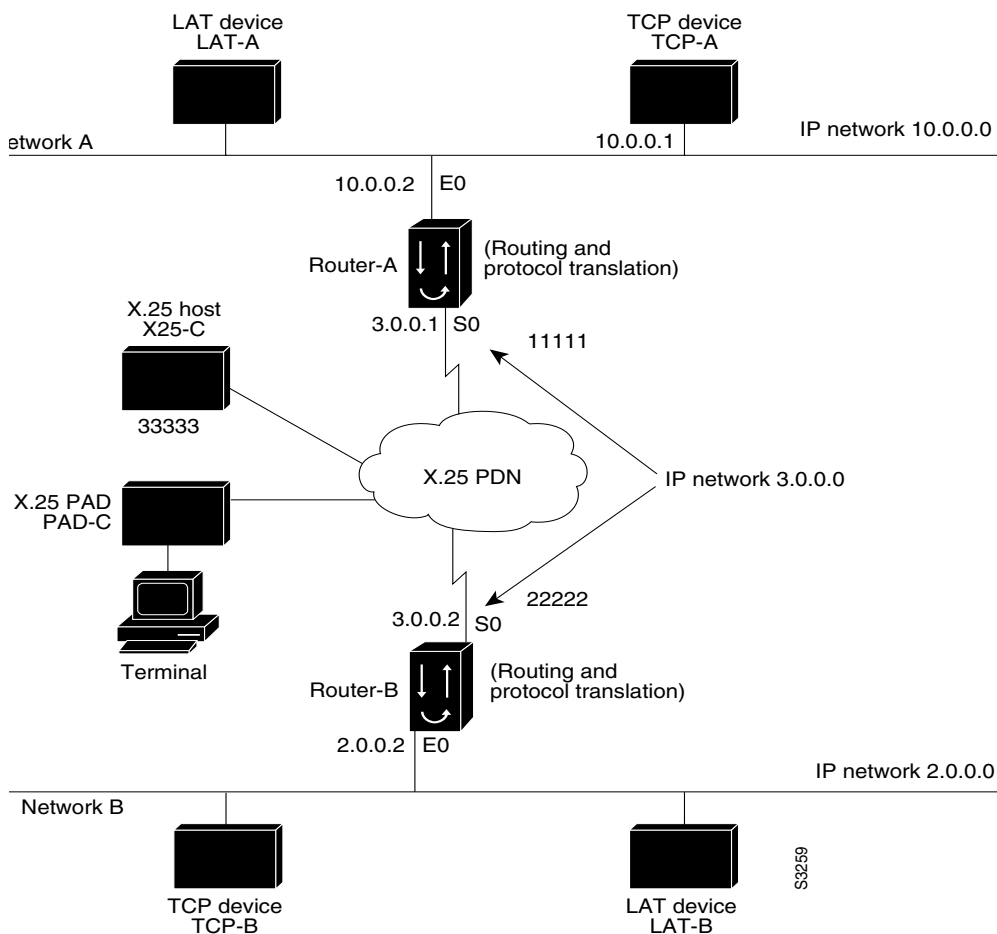
The following example shows how to configure the Cisco IOS software to assign an IP address dynamically to a PPP client using the one-step protocol translation facility.

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

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 43. Refer to the chapter “Configuring LAT,” in this publication, for more information about LAT. For information on configuring X.25, refer to the *Wide-Area Networking Configuration Guide*.

Figure 43 Diagram Showing Routers with Protocol Translation



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

Configuration for Router-A

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

```
interface ethernet 0
ip address 10.0.0.2 255.255.0.0
!
! Enable LAT on interface
lat enabled
!
interface serial 0
encapsulation X.25
x25 address 11111
!
! The following parameters may depend on your network
x25 facility packetsize 512 512
x25 facility windowsize 7 7
!
```

```

! IP address and MAP command needed only if routing IP
ip address 10.3.0.1 255.255.0.0
x25 map ip 10.4.0.2 22222 broadcast
!
! Set up IP routing
router igrp 100
network 10.0.0.0
network 10.3.0.0
!
! Advertise as available for connections via LAT
! Use this name (router-A) if connecting via 2-step method
! (for connecting directly to a specific router)
lat service router-A enable
!
! Set up some IP host names/addresses
ip host router-A 10.0.0.2 3.0.0.1
ip host TCP-A 10.0.0.1
ip host TCP-B 10.2.0.1
ip host router-B 10.3.0.2 2.0.0.2

```

Configuration for Router-B

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

```

interface ethernet 0
ip address 10.2.0.2 255.255.0.0
!
! enable LAT on interface
lat enabled
!
interface serial 0
encapsulation X.25
x25 address 22222
! The following parameters may depend on your network
x25 facility packetize 512 512
x25 facility windowize 7 7
!
! IP address and MAP command needed only if routing IP
ip address 10.3.0.2 255.255.0.0
x25 map ip 10.3.0.2 11111 broadcast
!
! Set up IP routing
router igrp 100
network 10.2.0.0
network 10.3.0.0
!
! advertise as available for connections via LAT
! Use this name (router-B) if connecting via 2-step method
! (for connecting directly to a specific router)
lat service router-B enable
!
! Set up some IP host names/addresses
ip host router-A 10.3.0.1 10.0.0.2
ip host TCP-A 10.0.0.1
ip host TCP-B 10.2.0.1
ip host router-B 10.2.0.2 10.3.0.2

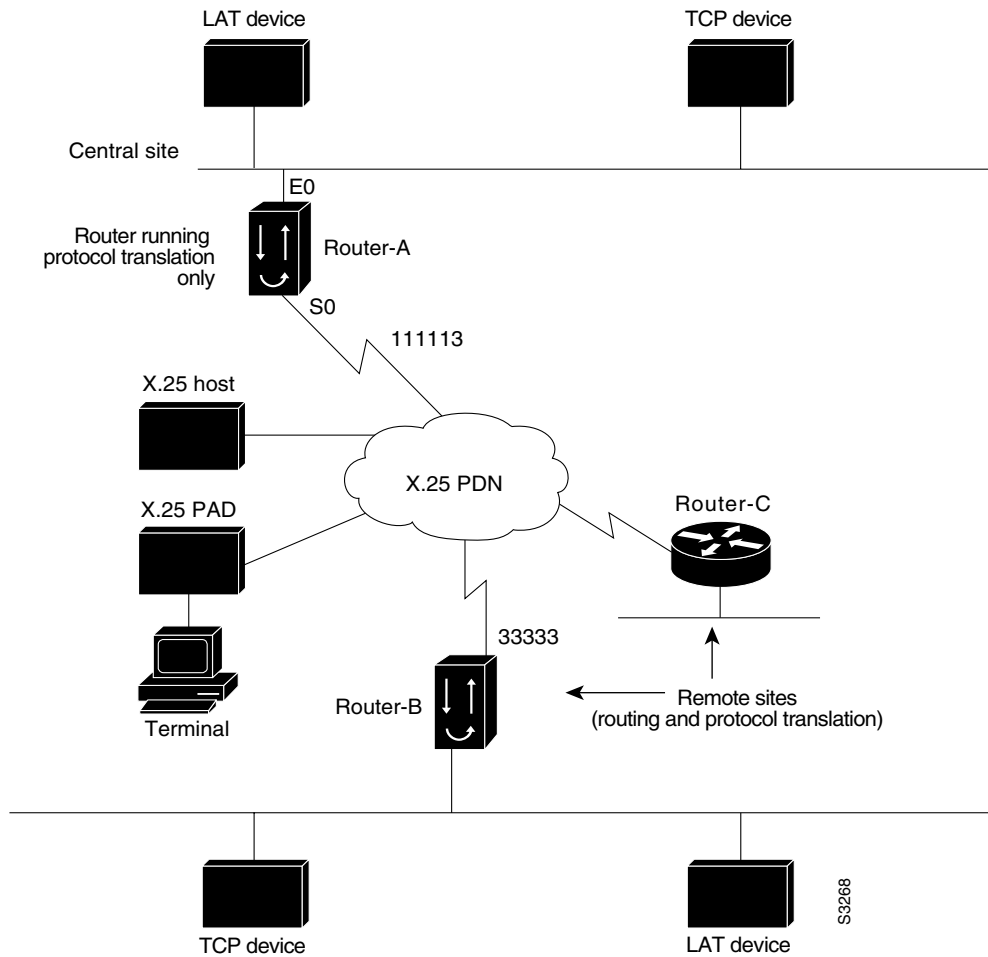
```

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

Central Site Protocol Translation Example

To support this application, a router with an image that supports protocol translation is directly connected back-to-back (Figure 44) to another router. 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 44 Central Site Protocol Translation Example



The following example illustrates how to configure a router to support translating protocols over an X.25 network among multiple sites.

Router-C is configured to act as an X.25 switch to send X.25 packets to Router-A while concurrently routing and bridging other protocols.

The following example also 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 10.0.0.2 255.255.0.0
 !
 ! enable LAT on interface if concurrently routing (8.3 feature)
 lat enable
 !
```

```

interface serial 0
  encapsulation X.25
  ! note that this is subaddress 3 of 11111
  x25 address 111113
  ! The following parameters may depend on your network
  x25 facility packetsize 512 512
  x25 facility windowsize 7 7
  no ip address

! "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 lat-device tcp tcp-device
! etc...any translate commands needed by application

```

Decreasing the Number of Translation Sessions Example

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

```
no line vty 10
```

Increasing the Number of Translation Sessions Example

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

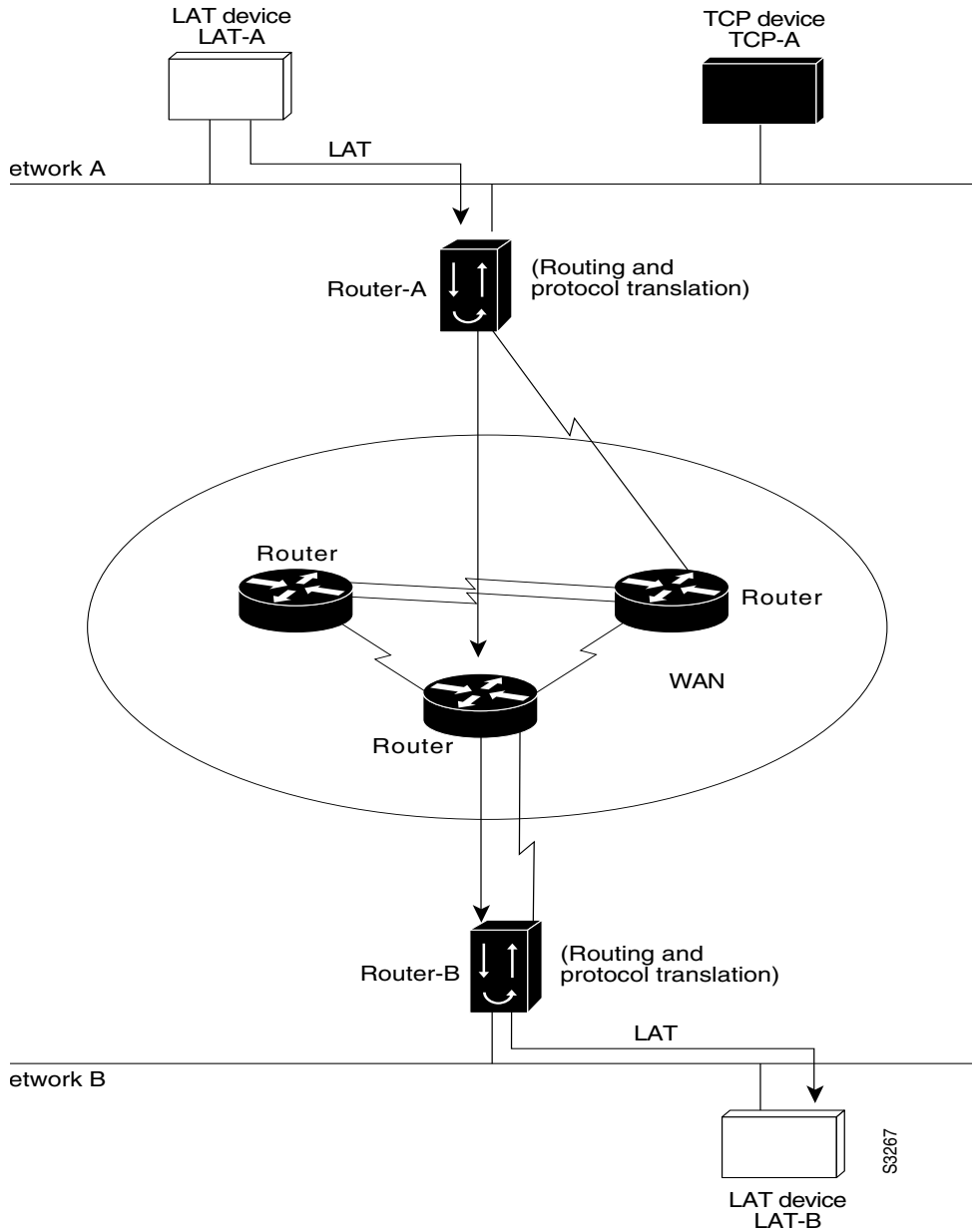
```
line vty 119
```

LAT-to-LAT over an IP WAN Example

The Cisco IOS software can be used to connect LAT devices over a WAN backbone that only allows routable protocols (see Figure 45). This configuration exists when LAT networks are either isolated or on their own internetwork.

With the protocol translation, LAT traffic can be translated to TCP and then routed on the WAN as TCP traffic. The LAT connections stay local between the LAT device and the router running the protocol translation option. Thus, connections are not susceptible to delays on the WAN. This 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 45 LAT-to-LAT over an IP WAN



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

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

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

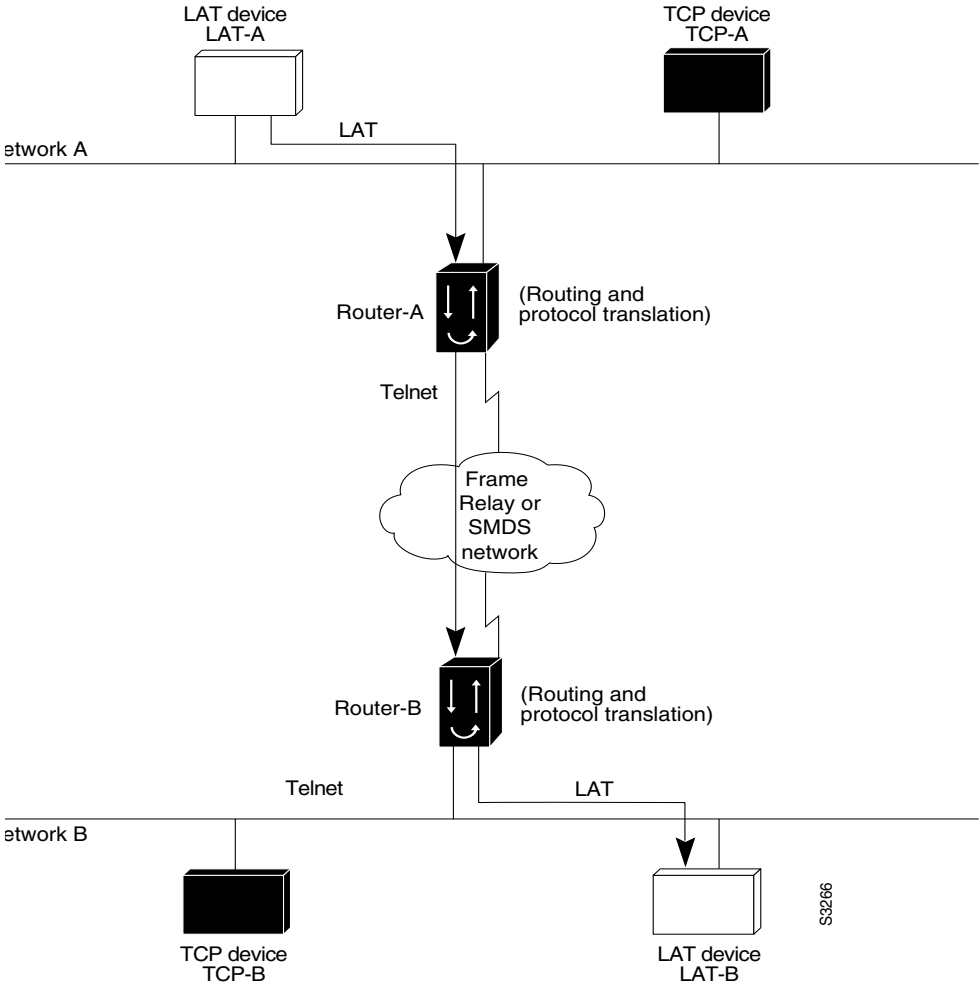
Note You can use the same name (for example, "LAT-B") in the **translate** command for both Router-A and Router-B, because each router operates independently. However, this symmetry is not required. The key is the common IP name in both **translate** commands.

LAT-to-LAT over Frame Relay or SMDS Example

To transport LAT traffic over a Frame Relay or an 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 46.

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 *Wide-Area Networking Configuration Guide*.

Figure 46 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 Cisco IOS software routes encapsulated packets translated from LAT to TCP over the Frame Relay or SMDS network. Packets are then translated back to LAT on the other side of the Frame Relay or SMDS network.

```

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

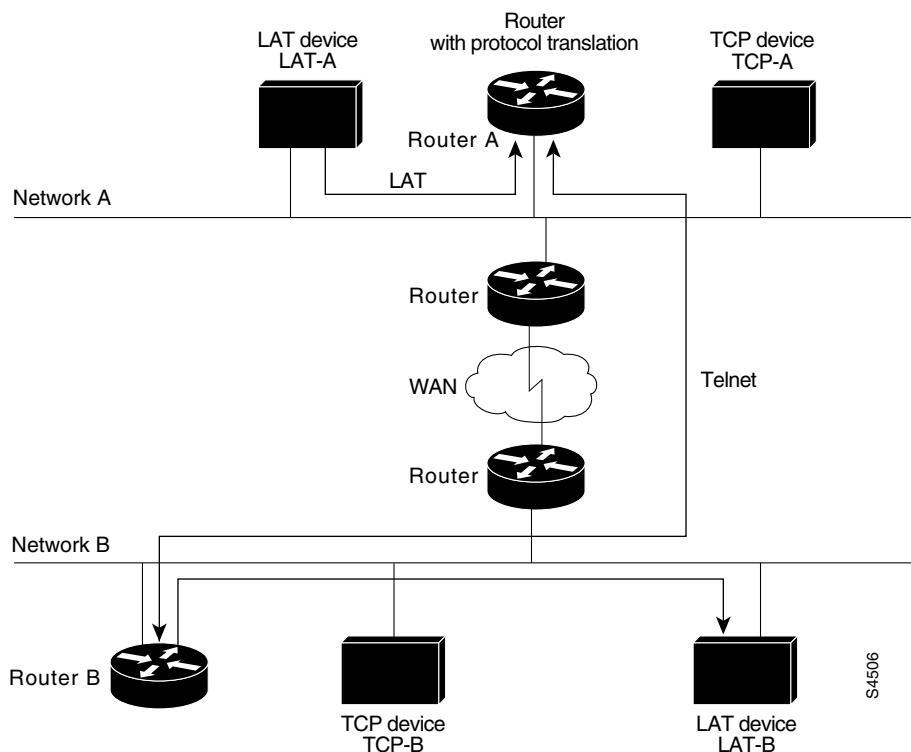
! Translate TCP to LAT on router-B, which is on Network B
translate tcp router-B lat LAT-B
    
```

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

LAT-to-LAT Translation over a WAN Example

In Figure 47, LAT can be transported to a remote LAT device by translating the packets to TCP format and using Telnet to send them across the WAN. The configuration files for Router-A and Router-B follow the figure. The logical name *CS-B1* is the name given to device *CS-B*.

Figure 47 LAT-to-LAT Translation over a WAN



Configuration for Router-A

```
interface ethernet 0
  ip address 192.168.32.16 255.255.0.0
  !
  ! enable LAT on this interface
  lat enabled
  !
  translate lat distant-LAT tcp TS-B1
```

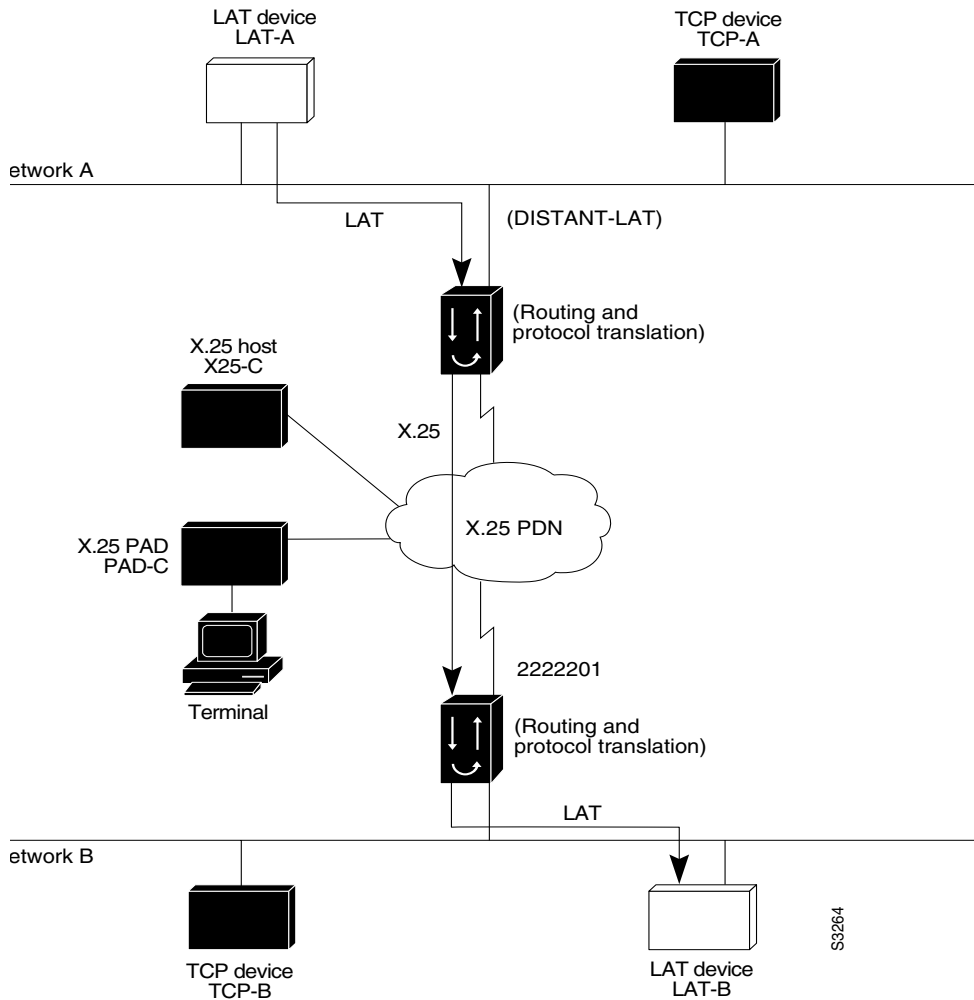
Configuration for Router-B

```
interface ethernet 0
  ip address 192.168.38.42 255.255.0.0
  !
  ! enable LAT on this interface
  lat enabled
  !
  translate lat TS-B1 lat LAT-B
```

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

Protocol translation provides transparent connectivity between LAT devices on different networks via an X.25 PDN. In Figure 48, 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 Cisco IOS software is configured for one-step protocol translation.

Figure 48 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 “Connection Commands” chapter of the *Access Services Command Reference*.)

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 48):

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

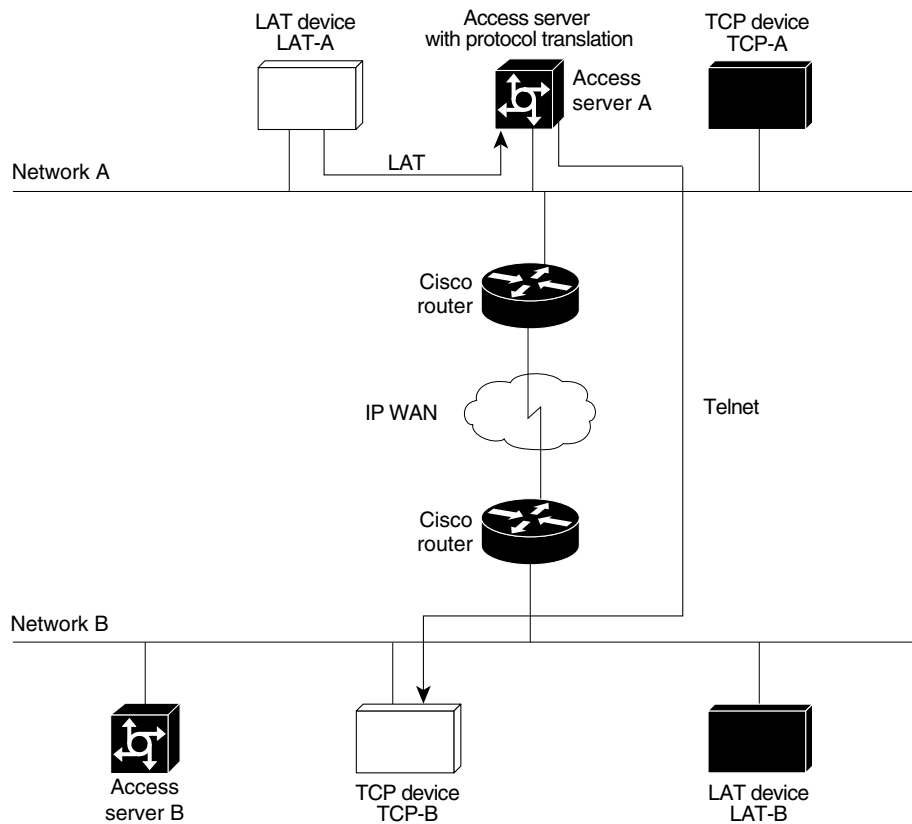
! Translate X.25 to LAT on router-A, which is on Network A
translate x25 1111103 lat LAT-A
```

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

LAT-to-TCP Translation over a WAN Example

Figure 49 shows a configuration that allows translation of LAT to TCP and transmission across an IP-based WAN. The configuration file for Router-A follows the figure. The logical LAT service name *distant-TCP* is the name given to device *TCP-B*.

Figure 49 LAT-to-TCP Translation over a WAN



Configuration for Access-Server A

```
interface ethernet 0
 ip address 192.168.38.42 255.255.0.0
 !
 ! enable LAT on this interface
 lat enabled
 !
 translate lat distant-TCP tcp TCP-B
```

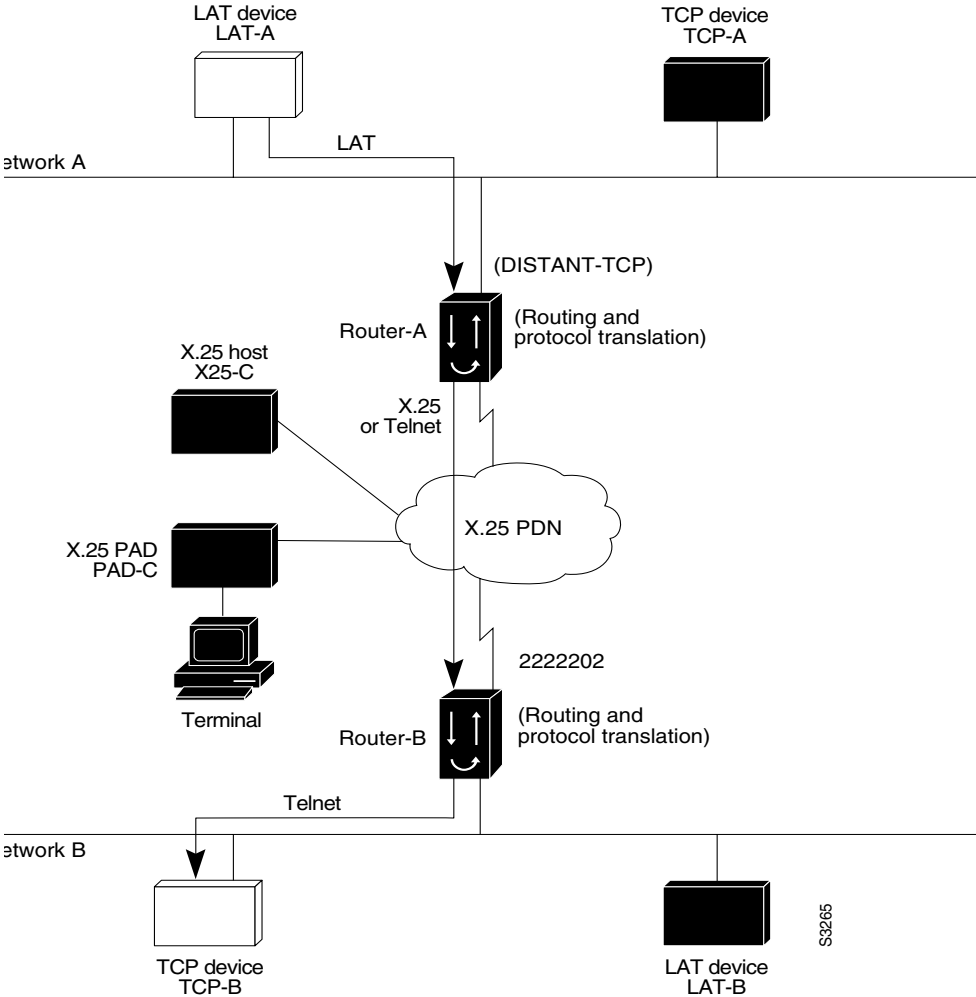
LAT-to-TCP over an X.25 Example

You can use protocol translation to provide transparent connectivity between LAT and TCP devices on different networks via an X.25 PDN. In Figure 50, which illustrates this application, the LAT device on Network A is communicating with the TCP device on Network B. There are two ways to provide this connectivity: the LAT traffic from Network A can be translated into either X.25 packets or TCP/IP packets 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 50 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 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.

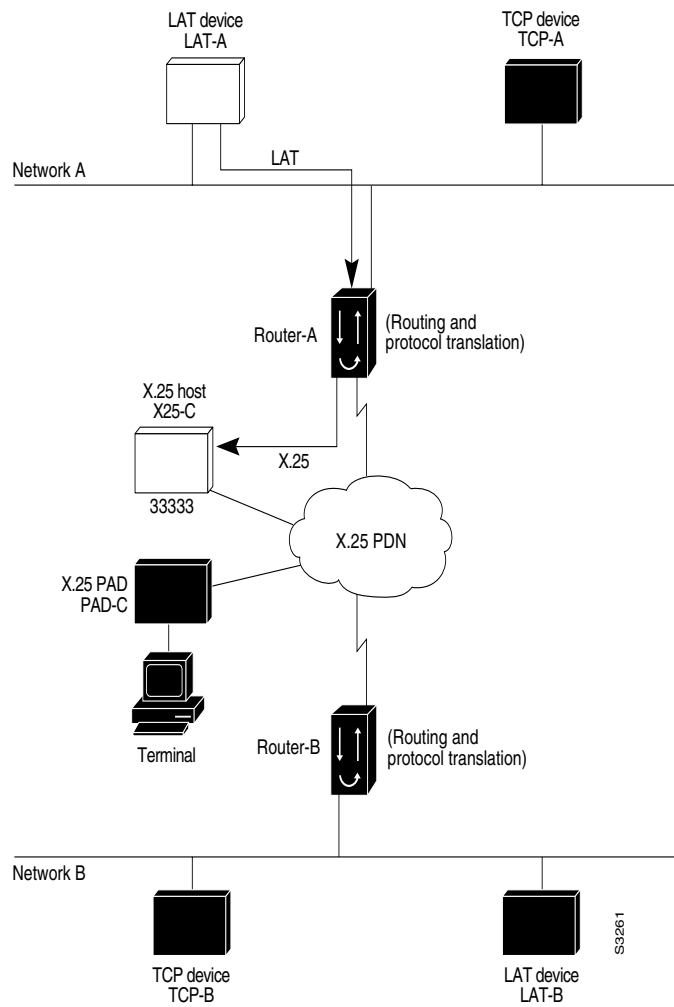
```
local> connect DISTANT-TCP
```

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

LAT-to-X.25 Host Example

Protocol translation permits LAT devices to communicate with X.25 hosts through an X.25 PDN. In the application illustrated in Figure 51, 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 51 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 with the protocol translation option to instruct the PDN to charge the destination for the connection. It is essentially a collect call. The reversal of charges must be prearranged with the PDN and destination location (on an administrative basis), or the call will not be accepted.

```

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

Local IP Address Pool Example

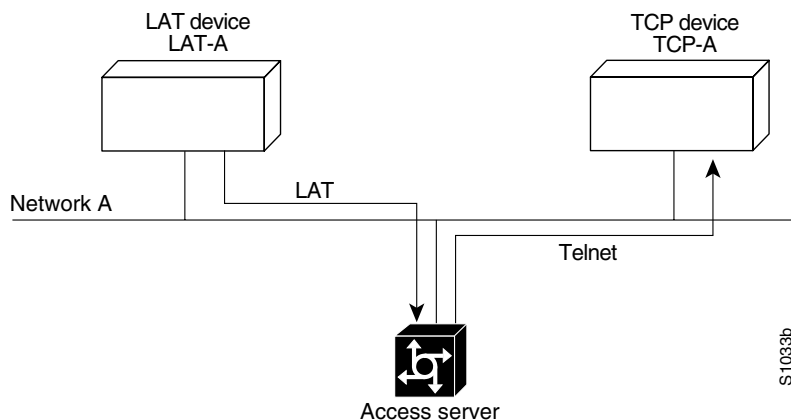
The following example shows how to select the IP pooling mechanism and how to create a pool of local IP addresses that are used when a client dials in on an asynchronous line. The address pool is named *group1* and comprised of interfaces 0 through 5.

```
! tell the server to use a local pool
ip address-pool local
! define the ip address pool
ip local-pool group1 192.168.35.1 192.168.35.5
translate x25 5467835 ppp ip-pool scope-name group1
```

Local LAT-to-TCP Translation Example

Figure 52 shows a simple LAT-to-TCP translation across an Ethernet network. Its Cisco IOS configuration file follows the figure. The name *TCPA* is the logical name given to the device *TCP-A*.

Figure 52 Local LAT-to-TCP Translation



Configuration for the Access Server

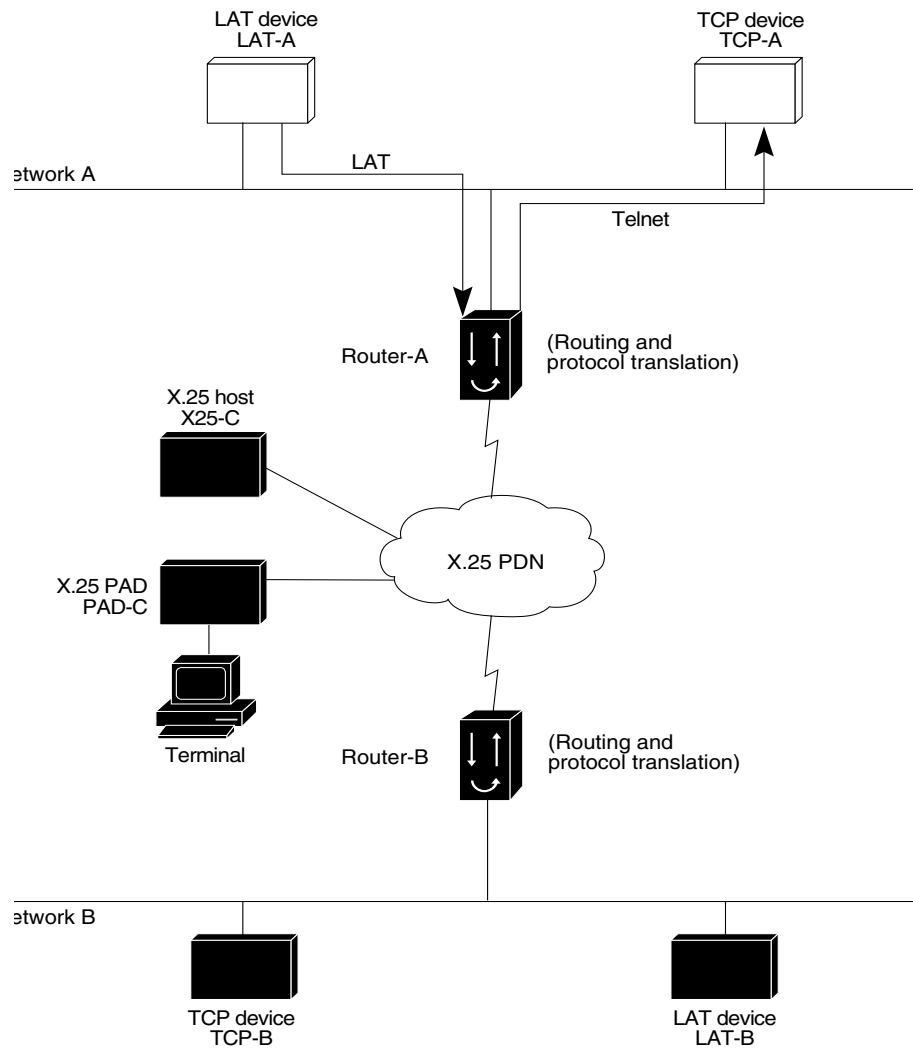
```
interface ethernet 0
 ip address 192.168.38.42 255.255.0.0
!
! enable LAT on this interface
 lat enabled
!
 translate lat TCPA tcp TCP-A
```

Local LAT-to-TCP Example

The Cisco IOS software running protocol translation can translate between LAT and Telnet traffic to allow communication among resources in these protocol environments. In Figure 53, 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 53 Local LAT-to-TCP Translation



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

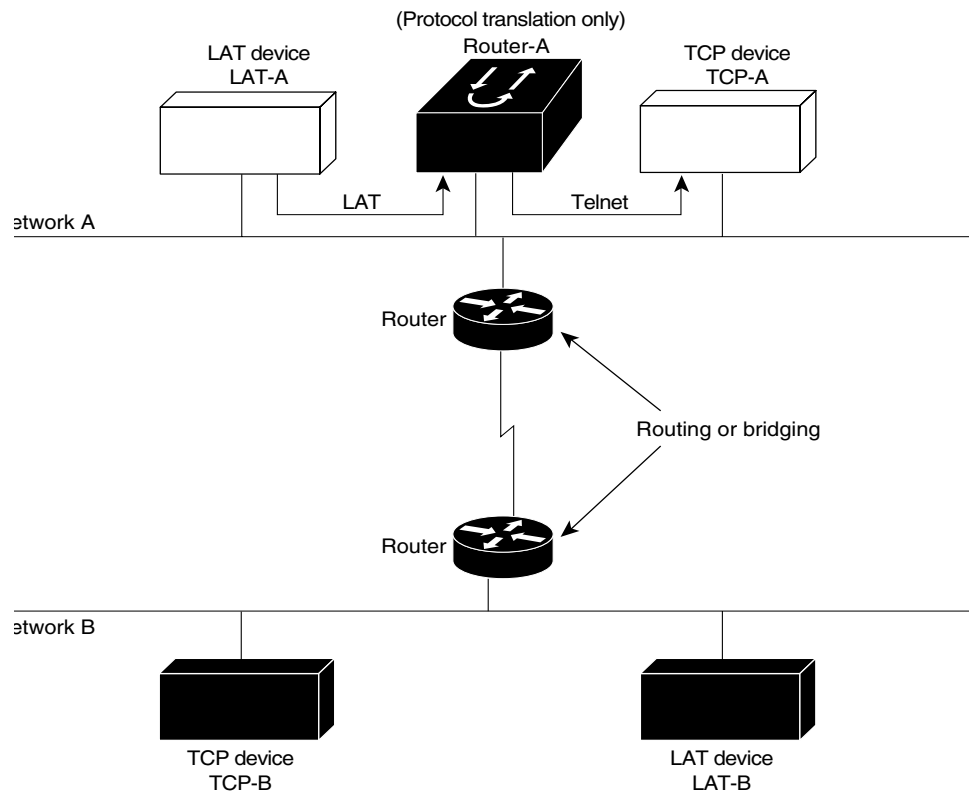
```
! Translate LAT connections to TCP for connectivity to TCP-A
translate lat TCP-A tcp TCP-A
! Optional additional commands
lat service TCP-A ident Protocol Translation to TCP-A
```

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

Standalone LAT-to-TCP Translation Example

If you need a large number of local LAT-to-TCP translation sessions, you can set up Router-A to use only an Ethernet port. This application allows 100 concurrent translation sessions. In the applications illustrated in Figure 54, any other router that supports protocol translation can be used to interconnect network segments performing bridging or routing.

Figure 54 Router Functioning as a Standalone Protocol Translator



S2377

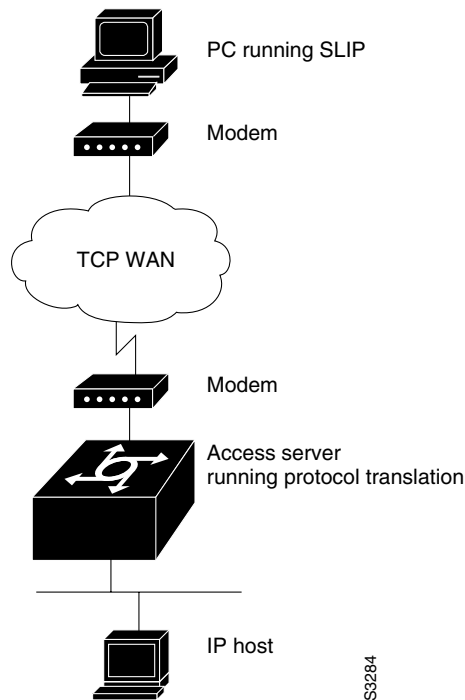
Configuration for Router-A

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

Tunneling SLIP inside TCP Example

Protocol translation enables you to tunnel from TCP to SLIP to allow communication among resources in these protocol environments. In Figure 55, 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 55 Tunneling SLIP inside TCP Example



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

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

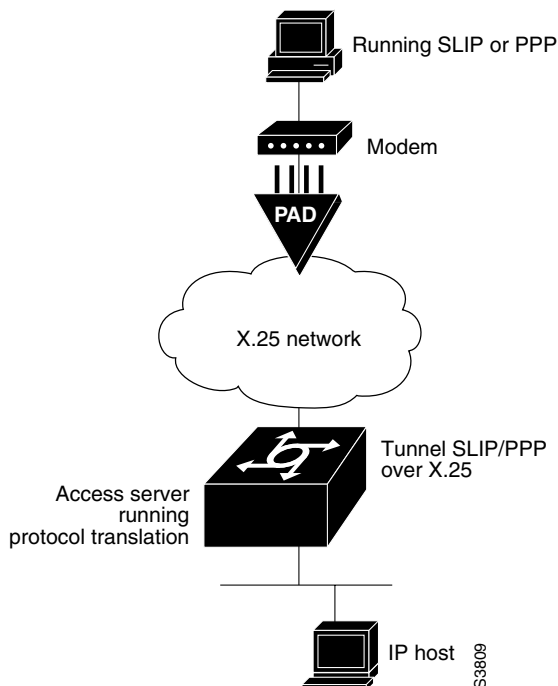
The device IP host on a different network attached to the router can be accessed by the SLIP client because routing has been enabled on the interface in the router where the SLIP session is established.

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

Tunneling PPP Over X.25 Example

Cisco IOS software can tunnel PPP traffic across an X.25 WAN to allow communication among resources in these protocol environments. In Figure 56, the PC establishes a dialup PPP session through an X.25 network using CHAP authentication.

Figure 56 Tunneling PPP in X.25



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

```
1:0, 2:0, 3:2, 4:1, 5:0, 6:0, 7:21, 8:0, 9:0, 10:0, 11:14, 12:0, 13:0, 14:0, 15:0, 16:127, 17:24, 18:18,
19:0, 20:0, 21:0, 22:0
```

For more information about X.3 PAD parameters, refer to the appendix “X.3 PAD Parameters” in the *Access Services Command Reference*. If you were performing a two-step connection, you would specify these X.3 PAD parameters using the **pad** [/profile name] command, which is described in the “Connection Commands” chapter of the *Access Services Command Reference*.

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

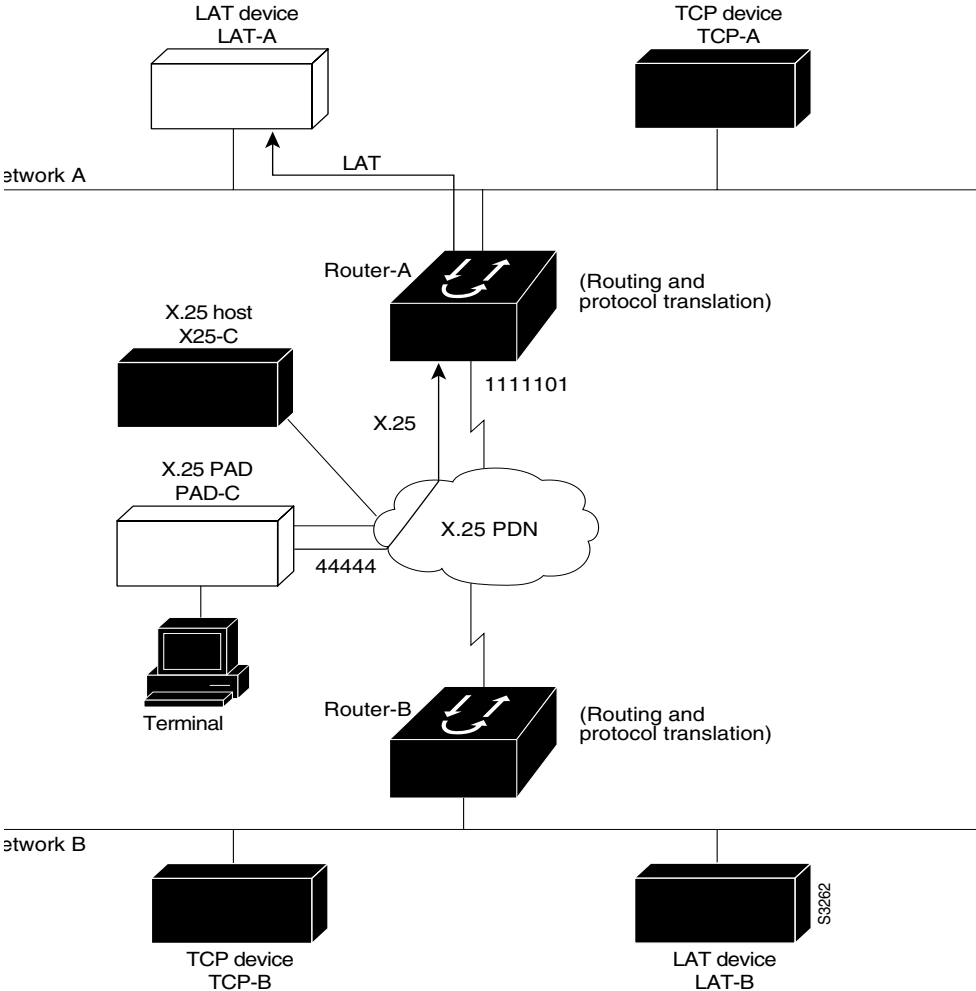
```
2509# config term
2509(config)# X29 profile x25-ppp 1:0 2:0 3:2 4:1 5:0 6:0 7:21 8:0 9:0
10:0 11:14 12:0 13:0 14:0 15:0 16:127 17:24 18:18
2509(config)# translate x25 31370054065 profile x25-ppp ppp 10.0.0.4 routing
authentication chap
```

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

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 57). X.25 PAD terminals make a call using an X.121 address, which is translated to a LAT node. To the PAD terminal user, the connection appears to be a direct connection to a host on the X.25 PDN. The Cisco IOS software also supports X.29 access lists, which allow you to restrict LAN resources (LAT or TCP) available to the PAD user.

Figure 57 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 Cisco IOS software. The first two lines define the scope of *access-list 1*. The first line specifies that access list 1 will permit all calls from X.121 address 44444. The caret symbol (^) specifies that the first number 4 is the beginning of the address number. Refer to the appendix “Regular Expressions” in the *Access Services Command Reference* for details concerning the use of special characters in defining X.121 addresses. The second line of the definition explicitly denies calls from any other number.

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

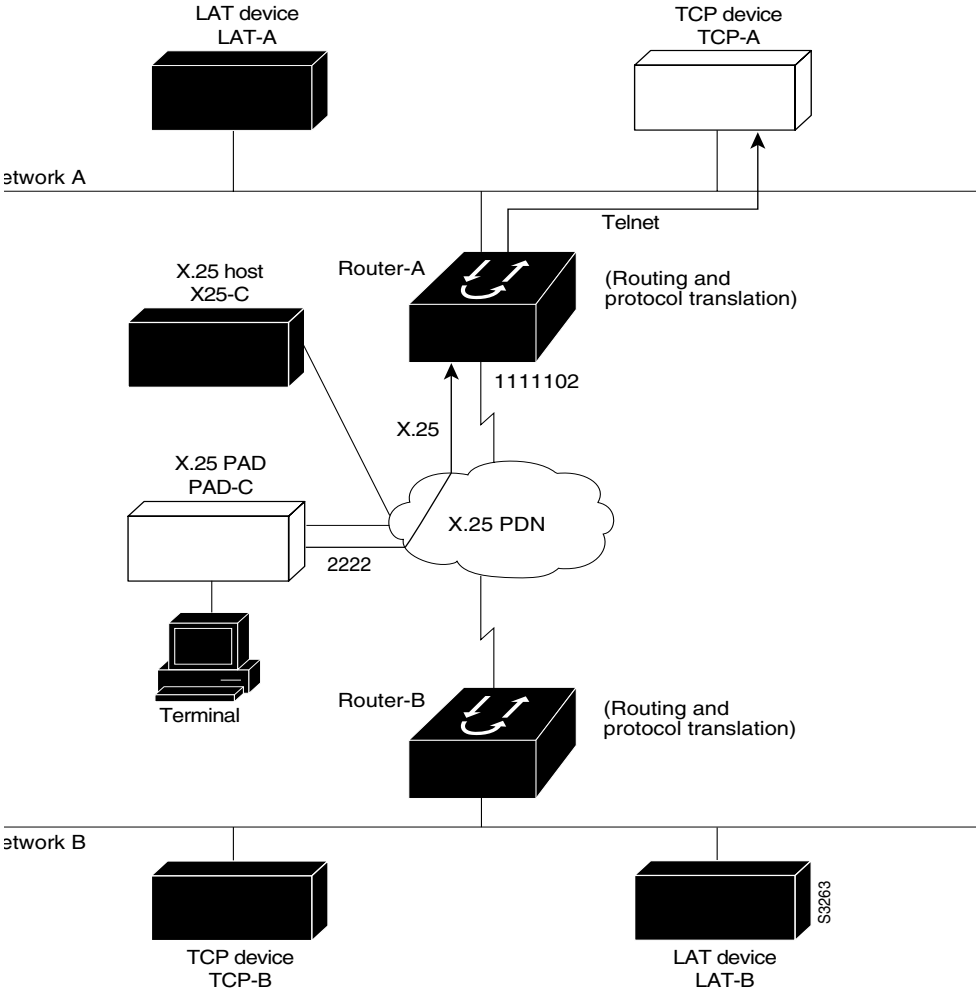
If you define multiple X.25 **translate** commands, each must contain a unique X.121 address. Also, the ITU-T protocol that transfers packets must match the X.121 addresses. This is specified in the protocol identification field of call-user data. This field specifies whether a packet is routed, translated, 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 with the protocol translation option. 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 58) is analogous to the preceding X.25 PAD-to-LAT example. Instead of translating to LAT, the configuration for Router-A includes a statement to translate to TCP (Telnet). Note that a router with the protocol translation software option can include statements supporting both translations (X.25 PAD-to-LAT and X.25 PAD-to-TCP). Different users on the same PAD can talk to X.25, LAT, or TCP devices.

Figure 58 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
```

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 172.16
  access-list 2 permit 172.16.0.0 0.0.255.255
  !
  !Permit LAT access to the prasad/gopala complexes.
  lat access-list 2 permit ^prasad$
  lat access-list 2 permit ^gopala$
  !
  !Permit X.25 connections to Infonet hosts only.
  x29 access-list 2 permit ^31370
  !
line vty 0 16
  access-class 2 out
  !
translate tcp 172.16.1.26 x25 5551234 access-class 2
```

X.3 Profile Example

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

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

The X.3 PAD parameters are described in the “X.3 PAD Parameters” appendix in the *Access Services Command Reference*.

Monitoring Protocol Translation Connections

The following example shows how to log significant VTY-asynchronous authentication information, such as the X.121 calling address, Call User Data (CUD), and the IP address assigned to a VTY asynchronous connection to a UNIX syslog server named *alice*:

```
service pt-vty-logging
logging alice
```