CHAPTER 19

Protocol Translation

This chapter describes how to make and configure protocol translation connections on the communication server. It assumes you understand how to use the configuration 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 system, LAT, Telnet, X.25, TN3270, and XRemote configuration chapters in this publication.

See the *Communication Server Command Reference* for a complete explanation of the **translate** command.

Cisco's Protocol Translation Methods

Protocol translation allows devices running dissimilar protocols to communicate. The translation of virtual terminal protocols allows connectivity to devices running different protocol stacks; however, there is no translation between other services such as file transfer protocols.

The protocol translator 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, LAT, and TCP/Telnet. Limited translation is supported for XRemote (to X.25 PAD environments) and TN3270 (to LAT, X.25, and TCP/Telnet environments). This 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: define a connection and the translation protocols in the configuration file, which is called the *one-step* method, or make actual connections between devices to the protocol translator, which is called the *two-step* method.

The One-Step Method

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

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

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

The Two-Step Method

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

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

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

Specifically, once the transmission protocols—LAT, X.25, TCP/IP—are configured, all that is required for the two-step protocol translation method is to enter the EXEC connection commands—**lat**, **pad**, **telnet**—first at one server, then onto another.

The protocol translator supports the two-step model in both directions (for example, from Telnet to PAD, and the reverse). See the end of the chapter for examples of two-step protocol translation sessions.

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

The two-step process also allows another level of security when TACACS and password protection is enabled. These security features are described in the system configuration chapter of this manual. 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 process, 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 TCP protocol translator. The difference is that you do not encounter the wiring complexity, unreliability, management problems, and performance bottlenecks that occur when two separate devices are connected via asynchronous serial lines.

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

One-Step Protocol Translation Configuration Task

To configure the one-step method of protocol translation, you use the **translate** command. This command sets 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—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.

- The connection features.

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.

Note The **translate** command does not support TN3270 or XRemote one-step translations. To use the protocol translator to convert these protocols, you must use the two-step method.

The task to create one-step protocol translation connection specifications is as follows:

Task	Command
Create the connection specifications for one-step protocol translation.	**translate** *protocol incoming-address* [*inoptions*] *protocol outgoing-address* [*outoptions*] [*global-options*]

See the *Communication Server Command Reference* for information about using this command. See the end of this chapter for examples of protocol translation configuration files.

Protocol Translation Application Examples

This section provides protocol translation examples for these applications:

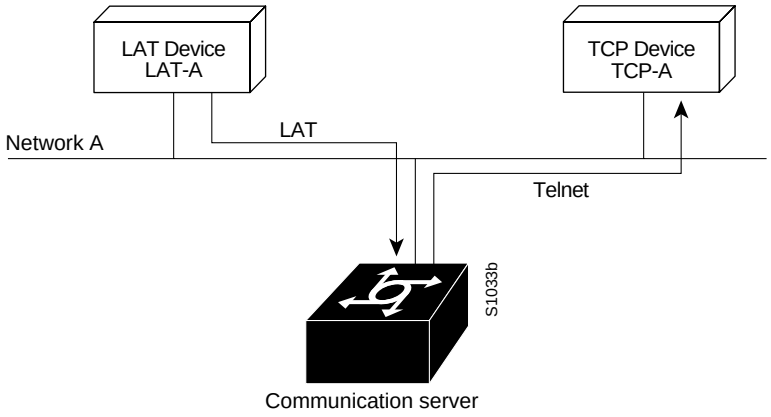
- Local LAT to TCP translations
- LAT to TCP over a wide area network (WAN)
- LAT to LAT over a WAN

Note In the application illustrations that follow, 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.

Local LAT to TCP Translation

This example, illustrated in Figure 1-1, shows a simple LAT to TCP translation across an Ethernet network. The name TCPA is the logical name given to device TCP-A.

Figure 1-1 Local LAT to TCP Translation



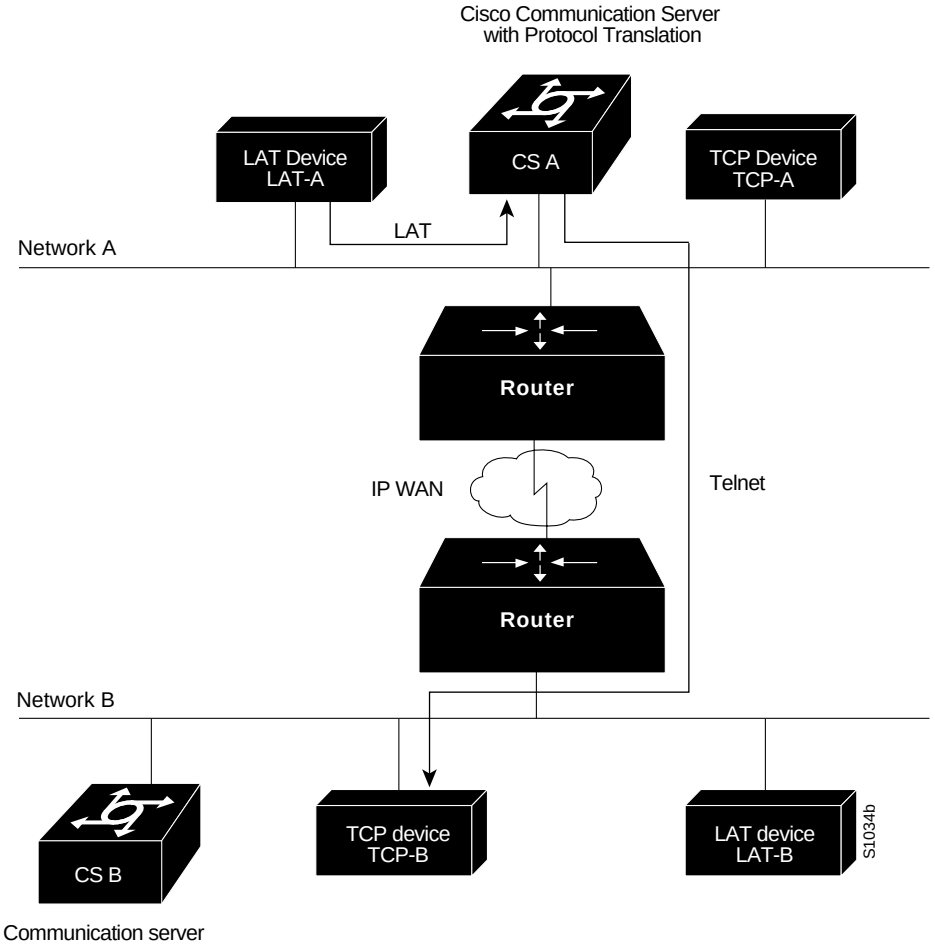
Configuration

```
interface ethernet 0
ip address 1.0.0.2 255.255.0.0
!
! enable LAT on this interface
lat enabled
!
translate lat TCPA tcp TCP-A
```

LAT to TCP over a WAN

This example, illustrated in Figure 1-2, shows a configuration that allows translation of LAT to TCP and transmission across an IP-based WAN. The logical name distant-TCP is the name given to device TCP-B.

Figure 1-2 LAT to TCP Translation over a WAN



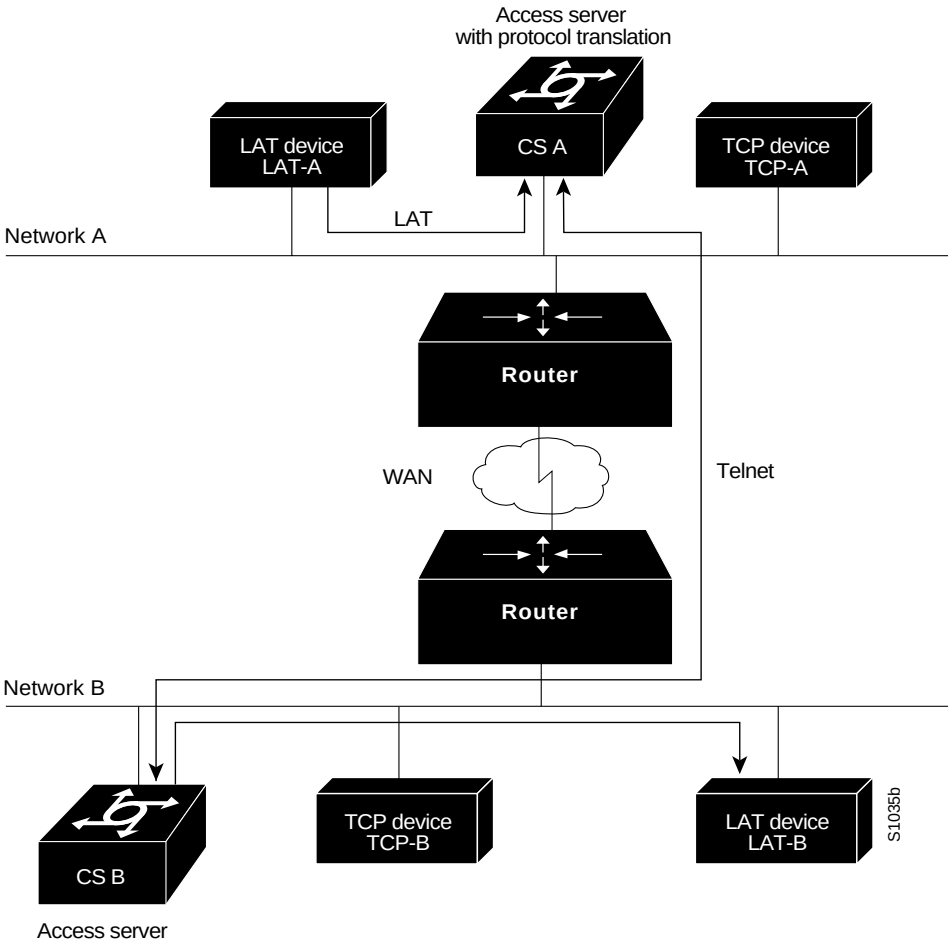
Configuration for CS-A

```
interface ethernet 0
ip address 1.0.0.2 255.255.0.0
!
! enable LAT on this interface
lat enabled
!
translate lat distant-TCP tcp TCP-B
```

LAT to LAT over a WAN

This configuration, illustrated in Figure 1-3, allows LAT to be transported to a remote LAT device by translating the packets to TCP format and using Telnet to send them across the WAN. The logical name *TS-B1* is the name given to device TS-B.

Figure 1-3 LAT to LAT Translation over a WAN



Configuration for CSA

```
interface ethernet 0
ip address 1.0.0.2 255.255.0.0
!
! enable LAT on this interface
lat enabled
!
translate lat distant-LAT tcp TS-B1
```

Configuration for CSB

```
interface ethernet 0
ip address 2.0.0.2 255.255.0.0
!
! enable LAT on this interface
lat enabled
!
translate lat TS-B1 lat LAT-B
```

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 protocol translator automatically converts the Telnet connection request to an X.25 connection request and transmits the request as specified in the system configuration.

- A connection is established by 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 protocol translator accepts the Telnet connection and immediately forms an outgoing connection with remote *host1* as defined in a **translate** command in your protocol translator's active configuration file.

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

The *host1* connection prompts for a user name, 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 protocol translator converts this to a Telnet option request on the UNIX host, which then stops the local echo mode.

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

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

Using the Two-Step Method for TCP to PAD Connections

The first step in any two-step translation is to connect directly from a terminal or workstation to a protocol translator.

Step 1 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 protocol translator named *orion* is accessible, it returns a login message and you enter your login name and password.

Step 2 The next step then is to connect from the protocol translator 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 protocol translator immediately sets the PAD to single character mode with local echoing, since this is the behavior the protocol translator expects. The PAD responds with its login messages and a prompt for a password:

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

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

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

Changing Parameters and Settings Dynamically

The following example illustrates how to make a dynamic change during a 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.

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

  ```
  Ctrl ^ x
  ```

- Then 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.

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

  The session resumes with the new settings.

But now you notice that information is not being displayed correctly. You may want to set the **/debug** switch to check that your parameter setting has not been changed by the host PAD.

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

  ```
  CS> 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.

- This example shows how to set ESC (ASCII character 27) as a dispatch character :

  ```
  CS> terminal dispatch-character 27
  ```

- To return to the PAD connection, simply enter:

  ```
  CS> resume
  ```