

Communications Server

User's Guide

601850401 Rev. B

Communications Server



1052 Melody Lane, Suite 210 • Roseville, CA 95678

For Technical assistance, call

(916) 783-1951

(916) 783-1952 Fax

Copyright © 1994 Penril Communications

A DIVISION OF RAYMAR INFORMATION TECHNOLOGY

The information in this manual is the property of Penril Communications and/or its suppliers and is protected under applicable copyright laws. No part of this manual may be reproduced or distributed without the express written permission of Penril Communications. Penril Communications reserves the right to alter the design and specifications of its products at any time without notice, as part of its continuing program of product development.

Before operating the Communications Server in the United States, you are advised to read the FCC guidelines in appendix F of this manual.

For service and warranty information, see the folder, *SERVICE INFORMATION AND WARRANTY*, which is included in the product shipping carton.

If you have a comment concerning this instruction manual, please send it to Penril Datability Networks, Technical Publications Group, 1300 Quince Orchard Blvd., Gaithersburg, MD 20878. Be sure to include the product name and manual part number (including revision letter) as printed on the title page. If your comment is about a specific page or pages, include the page number(s).



Specific product names, trademark names, etc., mentioned in this publication are the property of their respective owners and are mentioned for reference purposes only. These references do not constitute endorsement or nonendorsement by Penril Datability Networks.

Communications Server User's Guide

Table of Contents

Chapter 1 Introduction

Communications Server Features	1-2
Server Controls	1-5
Server Indicators	1-7
Server Line Cards	1-12
Obtaining Additional Information	1-15

Chapter 2 Configuring the Communications Server

Two Configuration Methods	2-1
Configuration Using the Configurator	2-2
Configuration Using the Parser	2-20

Chapter 3 Operating the Communications Server

Introduction	3-1
Running Startup Diagnostics	3-2
Executing Commands from Front Panel	3-8
Logging into Server through Parser	3-15
Session Control	3-18

Chapter 4 Managing the Communications Server

Managing the Server	4-2
File Transfer Between PCs and Hosts	4-4
Managing the IP Address Pool	4-8
Server Macro Facility	4-9
Storing Configuration Parameters	4-16
Using <i>SmartRAMCARD/S</i>	4-19
Storage/Retrieval Status Messages	4-26
Event Logger Messages	4-28
Software Uploading Options	4-29
Managing the Server Timeserver	4-42
Tracing a Route on the Network	4-46
Upgrading ENIC Software	4-50
Activating the UNIX Command Set	4-52

Appendix A Glossary of Variables	A-1
Appendix B Default Keyboard Mapping Profiles and Worksheets	B-1
Appendix C IP-Over-X.25 Communications	
IP-over-X.25 Communications	C-2
Getting Started with IP-Over-X.25	C-7
Appendix D UNIX/DECserver Equivalents	D-1
Appendix E Error Messages	E-1
Appendix F Notices	F-1

Introduction

Use this manual in conjunction with the *Communications Server Installation and Service Guide*.

Organization

The *Communications Server User's Guide* contains instructions for configuring, operating and managing the Communications Server.

Chapter 1 of this *User's Guide* provides an overview of the features and hardware for the Communications Server. The Communications Server is the first modular multipurpose server to combine both wide area and local area networking functionality in a cost-effective, high-performance design. The Communications Server features an Enhanced Network Interface Card (ENIC) offering either single- or dual-protocol networking capabilities, and a selection of interchangeable line cards with a variety of wide area network (WAN) gateway and asynchronous communications options.

This chapter includes—

- Software features
- Server controls
- Server indicators
- Server line cards
- Additional documentation available
- Platform and ENIC specifications

Communications Server Features

Multiprotocol support includes TN3270 communications and Novell IPX printing, as well as sophisticated implementations of LAT and TCP/IP.

Newer software features are summarized below in the section on *ENIC^{Plus} Software Features*. For more information about using the Communications Server, see the *Communications Server Installation and Service Guide*.

ENIC^{Plus} Software Features

Installing the new *ENIC^{Plus}* in the server activates the software features. Additional *ENIC^{Plus}* software features (such as IP-over-X.25 communications), are only activated when you install an optional line card. For complete line card installation and operation instructions, see the user's guide that accompanies the line card.

Some of the features listed here are optional and only are included in the *ENIC^{Plus}* for an additional charge. All of these features and their functions are described in greater detail in this manual.

Protocol-Related Features

BOOTP protocol—Allows the server to obtain both a temporary IP address and the name of a file containing both a start-up and an operational image to load into internal memory. The source of the image can be another host on the network. Bootp can also retain a list of IP addresses that are accessible for use by other devices, such as routers, to determine their own IP addresses.

Dual protocol LAT-TCP/IP network compatibility—Forward and reverse LAT or Telnet operation over an Ethernet includes NCP and TSM or Penril Datability Networks' own server management utility, *VSM*. Reverse LAT supports incoming Host-initiated Requests (HIRs) to the local server. Reverse Telnet supports incoming HIRs when the local server acts as a Telnet host.

Finger command—Execute the **Finger** command to produce displays of information about users logged into a particular local or remote network.

IP-over-X.25—The IP-over-X.25 packet encapsulation software on the X.25 PAD Card expands the X.25 connectivity structure to users of IP. It permits the transmission of IP data packets over X.25 Wide Area Networks (WANs).

IPX printing support—Enables Novell users running IPX software to access network printers that are attached to the server.

PAP/CHAP—Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP) provide increased security across a PPP link.

PPP support—Point-to-Point Protocol (PPP) operates over a serial link. You can create and assign a PPP link to any server port. PPP links can be created using either the configurator or the command parser.

RIP support—Routing Information Protocol (RIP) is an Ethernet gateway protocol, that provides “reachability” information to devices on the LAN via continuous broadcasting of routing information. Devices using RIP know the latest routing information on the network.

SLIP support—Serial Line Internet Protocol (SLIP) permits users or host processors to log into hosts on the Ethernet and exchange data with devices that either cannot be connected directly to the Ethernet, or that reside on an Ethernet other than the one in which the source user resides. SLIP lines are point-to-point serial connections running TCP/IP. SLIP links can be created with the configurator, or manually through the command parser.

Dynamic SLIP support—Permits users to log into a server port and execute the **Connect slip** command to automatically reconfigure that port as a SLIP serial interface. Thereafter, the server acts as a Telnet IP gateway between the user’s host and the IP network. A major advantage of using dynamic SLIP is that the same port that is physically connected to the SLIP line also automatically executes the IP setup commands that are needed to start and maintain the connection.

SNMP (Simple Network Management Protocol)—This TCP/IP domain management function gives an administrator (working from a Network Management Station (NMS)) an overview of network status and direct control over network devices. The server will function as an agent that sends information to the NMS in response either to a formal request or to a status change or event (e.g., a server reboot).

TFTP (Trivial File Transfer Protocol)—Can load start-up and operational images to and from the server. A server can also get an image file from a second server if the target server knows both its own IP address and the IP address of the source server.

TN3270 product support (optional)—TN3270 is a subset of the TCP suite of protocols designed to allow TCP/IP users to connect to IBM hosts physically connected to the Ethernet. Using a VT terminal, one can connect via Telnet to an IBM host to access a particular application.

Traceroute—This is a debugging tool. If users have difficulty connecting to a target host, you can execute commands that trace the route a packet takes from the source server to the destination host.

UNIX Features

UNIX command set—The server accepts entry of a subset of UNIX commands, which it translates into equivalent parser functions. See appendix D for a complete list of UNIX-executable commands and their DECserver (or command line) equivalents.

Rlogin—Simplifies the login process by allowing users to log into UNIX-based Telnet hosts without the additional step of entering a username or a password. Rlogin protocol is started by executing the **Connect Rlogin** command.

LP printing support—Enables UNIX users to send files to network printers attached to the server.

Other Software Features

Centralized Account Reporting System (CARS)—CARS is an audit trail data base of event log messages, session information and login information. Data can be directed to a port or saved on *SmartRAMCARDS*.

Configurator function—Use this simple, question-driven interface to set up many of the server operating parameters, including IP links, SLIP link setup and PPP link setup. On-screen help explains the purpose of each configuration parameter and lists the available choices.

Macro menu facility—Writes short “C” scripts to create customized interfaces or to execute server commands at user ports. Macro programs contain server commands, variables and compound statements.

Modem control selectivity—Server users can select either a CTS/DSR input combination, which activates RTS/CTS flow control, or an RI/DSR combination, which activates modem-ring handshaking. The switch is done using a **Change Port Handshake** command.

On-line help—Complete, interactive on-line help is available to logged-in server users. The on-line help describes each of the executable parser commands. A special tutorial feature describes how to use commands in commonly executed procedures.

SmartRAMCARD software uploading/downloading—A slot in the ENIC (on the rear of the Communications Server) accepts insertion of a *SmartRAMCARD*. These cards can contain software upgrades or diagnostics, that can be uploaded to the server. Files can be saved to the *SmartRAMCARD*.

Software exporting—The servers can supply *start-up* and *operating* images to other servers on the network. For example, if one updates an ENIC with the newest version of software, that updated server can supply a copy of its software image to all other ENICs on the network, if desired. The software exporting feature allows one to properly manage the upload process and to avoid discharging copies of the source image at random.

Telecommuting—Servers can provide laptop- and notebook-computer users access to LAT and TCP/IP hosts (at corporate headquarters, for example) via modem communication and PPP or SLIP. You can operate a laptop PC “from the road” as though that PC were tied directly into the regional office’s LAN.

Timeserver command—A built-in clock can synchronize other network servers for precisely timed execution of server maintenance and administrative functions. You can document system messages and activities with the date and time.

Upgrading server software via password keys—You can easily upgrade your server software as follows:

- You can convert a single-protocol (LAT-only or TCP-only) server to a dual-protocol (LAT-TCP/IP) server.
- You can add TN3270 protocol support to a server.
- You can add multiple server download/site feature to a server.

Server Controls

Server controls are touch type on front and back panels (figure 1-1). Controls include—

- Alphanumeric keys
- Cursor control keys to execute diagnostic routines and operational commands
- **ENTER** and **DELETE** keys

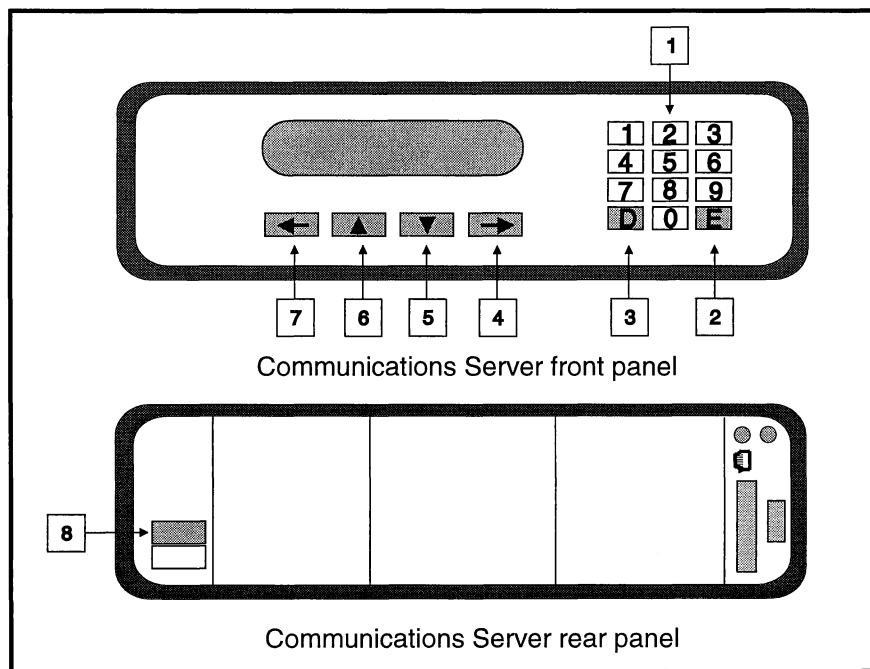


Figure 1-1. Communications Server front and rear

- Power **I/O** (on/off) switch, which controls the flow of electricity to the server to either start or clear-and-restart the server

See table 1-1 for summary descriptions of each control.

Table 1-1. Summary of Control Key Functions

No.	Control key	Key functions
1	1ABC, 2DEF, 3GHI, 4JKL, 5MNO, 6PQR, 7STU, 8VWX, 9YZ., 0-	Press numerals to select commands. Press alphabetic characters to execute keywords within commands. Use the period to separate the bits in IP addresses. Use the hyphen to separate the bytes of Ethernet addresses. Use the underscore to join two words into a single string, such as Host_name.
2	DELETE	Press to cancel the current instruction or command character.
3	ENTER	Press to execute the current instruction or command.
4	→	Cursor right. Press to move forward within the list of command options.
5	↓	Cursor down. Records the character at the cursor location and advances the cursor to the next command parameter.
6	↑	Cursor up. Press to move from the current level in the command hierarchy to the previous level.
7	←	Cursor left. Press to move backward within the list of command options and to exit the list.
8	I/O	Powers the server on and off.

Alphanumeric Keys

Execute server commands from the front panel by pressing the appropriate alphanumeric key. Because the front keypad has no **Num Lock** key to select alpha characters or numerals, when such a selection must be made, repeatedly press the key to step through each of the available characters until the desired character is obtained. For example, to obtain a **C**, repeatedly press the **1ABC** key until the **C** appears in the display.

Cursor Control Keys

The [→] key enters a displayed character and moves the cursor to the next entry position. Repeat for each of the required character entries. Press the cursor control keys to move from one level of the command hierarchy to another. For example, after you have entered the number for the **Set** command, press the cursor down [↓] key. The LCD screen displays options that may be executed in conjunction with the **Set** command. The cursor keys also permit you to display different portions of the LCD screen.

ENTER and DELETE Keys

When all of the required characters are displayed, press the **ENTER** key to save the accumulated entries for server execution. This signifies the end of a command string and downloads the command to the server. If, during the execution of a command, you execute an incorrect character, use the **DELETE** key to remove the incorrect character from the command string. The **DELETE** key must be one space to the right of the character to be removed.

Power Switch

The switch is located on the back panel of the server, above the power cable socket. Use this switch to power the server either on or off. To start or to clear-and-restart the server, press the rocker switch so the “P” is displayed. This starts the flow of electricity to the server. Power off the server by pressing the switch so the “O” is displayed.

Server Indicators

The Communications Server uses indicator lights on the back panel and an alphanumeric display screen on the front panel to communicate information about internal operations and the state of the Ethernet. Such information includes network activity, command entries, and server messages. Server indicators are the:

- Liquid Crystal Display (LCD) screen
- Twisted-pair-link-status-indicator Light Emitting Diode (LED)
- Network-activity-and-status-indicator LED

Following are descriptions of the server's onboard display screen and indicator lights. Figure 1-2 shows the location of each item on the front and rear panels of an ENIC-equipped Communications Server.

Table 1-2 describes each of the indicators in figure 1-2. Additional descriptions follow the table.

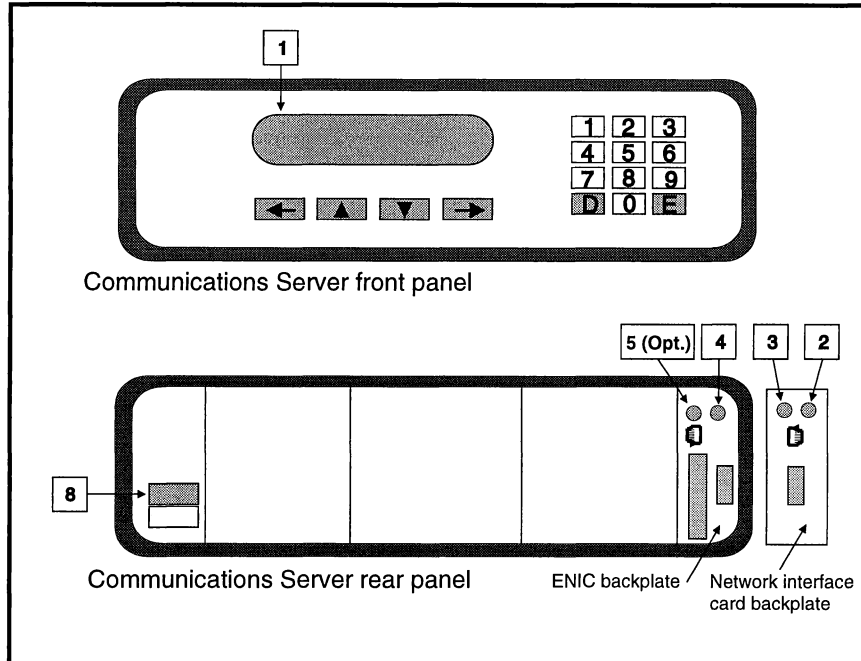


Figure 1-2. ENIC-equipped VCX Gateway server

Table 1-2. Indicator Functions

No.	Indicator	Function
1	Liquid Crystal Display Screen	Displays command entries, server messages and information about diagnostic routines.
2	Twisted Pair Link Status LED (<i>green or off</i>) (<i>NIC only</i>)	Steady green glow indicates a good Ethernet link. Off indicates that there is no Ethernet link. Only operates if using twisted pair wiring.
3	Network Interface LED (<i>red or off</i>) (<i>NIC only</i>)	Flickers red with network activity. Off indicates that there is no Ethernet link.

No.	Indicator	Function
4	Network Interface LED (green, red, or off) (ENIAC only)	Flashing green indicates normal Ethernet traffic. Flashing red indicates that network collisions are occurring. Steady red glow indicates that the thinwire cable has been removed. <i>Off</i> indicates that there is no Ethernet traffic.
5	Twisted Pair Link Status LED (optional) (green or off) (10Base-T-equipped ENIAC only)	Steady green glow indicates an established link is operating properly. <i>Off</i> indicates that the link either has not been properly established or has been disconnected.

The LCD Screen

The server front panel features an LCD screen. The screen displays four lines of text and measures 40 characters across. The LCD screen is backlit by an electroluminescent light source that enhances the contrast for improved visibility. See the table 1-1 for descriptions of item numbers.

Use the onboard LCD screen to observe command entries, server messages and server diagnostic routines. By working at the server front panel, you see immediate feedback from that server about its configuration, physical installation and network communications. Execute commands from the server front panel to configure the software so that the port, server and service characteristics match those of the attached devices. Server messages can be displayed to show the results of the executed commands. Server diagnostic routines are only accessible from the front panel.

Operating the LCD Display

The screen will not light if an ENIC is not installed in the server. If the ENIC is installed in the server, the Ethernet cabling need not be connected to the ENIC for the screen to light.

The light source operates only when a status message is received at the server or when a key is pressed at the front panel keypad. It remains lit for *three minutes* without additional activity at the server front panel. The information remains on the screen indefinitely; however, the server only retains up to twenty-one lines of text. When the twenty-second line of text is received, the first line disappears from the screen and cannot be retrieved at the front panel. To retrieve earlier messages, log into a terminal at a server port and execute Show Events.

If the screen is dark and the depressed key cannot change the display, the screen remains dark. For example, if the server screen displays the end of the command menu (Start, Stop, Test, Zero) and you press the [→] key to view the next display, since there is no following display, the screen stays dark.

LCD Screen Viewing Area

Although a full-view display (such as the screen on a standard-sized CRT monitor) provides one complete picture that is most convenient to read, the entire contents of a full-view display also can be read using the smaller LCD screen. Typically, a standard-sized CRT monitor measures 24 lines by 80 characters, while, by contrast, the Communications Server's LCD screen measures just four lines by 40 characters. You can make this comparison if you place the LCD screen in the upper left-hand corner of the full-view display. Note that the viewing area of the LCD screen covers only a small region of the full-view, CRT display. In fact, the LCD screen occupies $\frac{1}{12}$ th of the full-sized display. See Figure 1-3 and the formula below. (FVD stands for the full-view display and LVD for the limited-view display.)

$$\left[\frac{24 \text{ lines}}{4 \text{ lines}} \frac{\text{FVD}}{\text{LVD}} \right] \times \left[\frac{80 \text{ characters}}{40 \text{ characters}} \frac{\text{FVD}}{\text{LVD}} \right] = 12 \text{ regions}$$

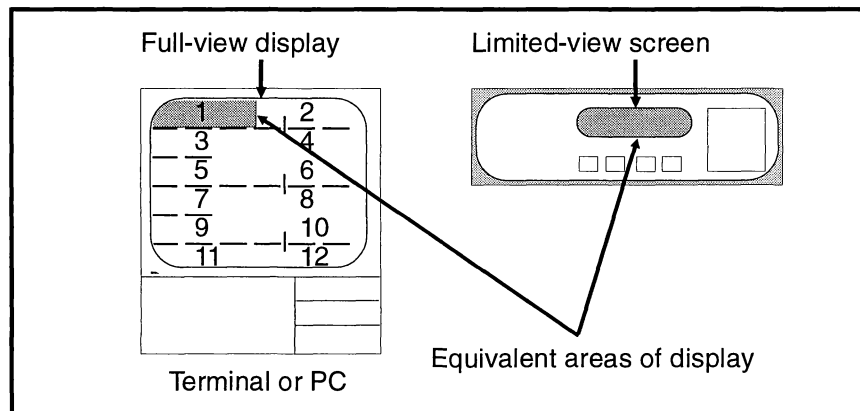


Figure 1-3. Limited and LCD view

Locating LCD Viewing Area on Full-view Display

You can determine the exact area of coverage by referring to the semaphore in the subscreen on the LCD display. The subscreen is a rectangle located in the upper right-most portion of the LCD screen. The subscreen contains a small black rectangle called a semaphore. This marks the region of the full-view display currently shown on the limited-viewing area of the LCD screen. See figure 1-4.

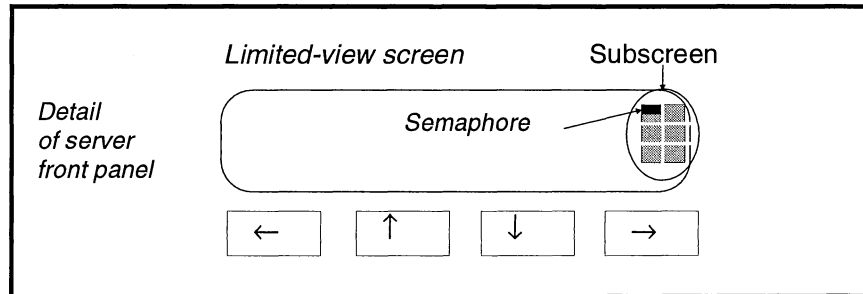


Figure 1-4. Subscreen and semaphore

To look at a different part of the full-view display, press the appropriate arrow keys to move the semaphore in the subscreen until you mark the desired region of the full-view display.

The semaphore only moves either horizontally or vertically, one region at a time. To move across several regions, repeatedly press the required cursor control keys. See figure 1-5.

Disabling the LCD Screen

If the convenience of the front panel controls is not required in your installation, you can disable the onboard display by executing the **Set Panel Disable** command.

Twisted-Pair-Link-Status-Indicator LED

The twisted-pair-link-status-indicator LED only appears on ENICs with the optional 10Base-T adapter and operates only if the server is connected to the Ethernet using twisted pair wiring. A steady green light indicates a good Ethernet link; no light indicates no link.

Network Interface LED

The network interface LED appears on all ENICs, and it indicates the status of data traffic between the ENIC and the Ethernet. Bipolar (red/green) LEDs are used on ENICs to indicate different levels of network activity. Older style NICs use red LEDs. For complete descriptions of the LED functions, see the table beneath figure 1-2.

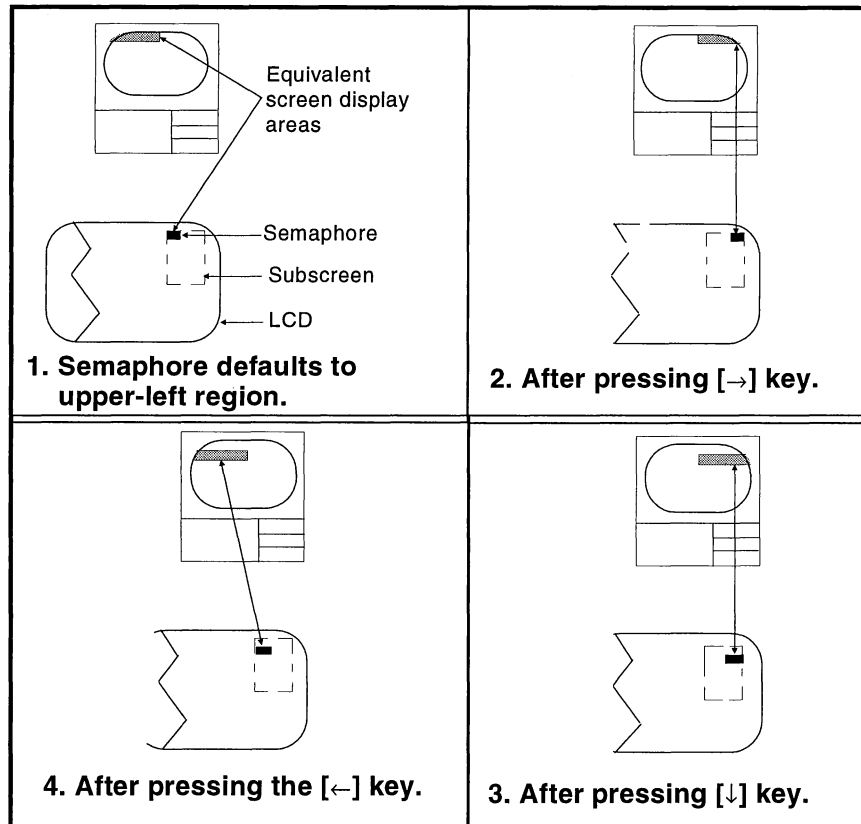


Figure 1-5. LCD location versus semaphore indicator

Server Line Cards

The Communications Server includes both an Enhanced Network Interface Card, offering either single-protocol or multiprotocol networking capabilities, and a selection of interchangeable line cards, which provide a suite of wide area network gateway and asynchronous communications options. By combining these components, you can customize the Communications Server to fulfill multiple networking and connectivity needs—from a terminal server within a single-protocol network, to a multisession communications server operating in multiprotocol environments; providing both terminal server and gateway capabilities.

The Communications Server is designed so new capabilities can be added as the network grows—by adding or replacing line cards. Each line card features a microprocessor to ensure the greatest possible network speed, and comes with detachable port connector plate, and can be installed in seconds. The line cards

can be swapped in and out of the Communications Server quickly and easily, as all configuration parameters are maintained in battery-backed-up memory.

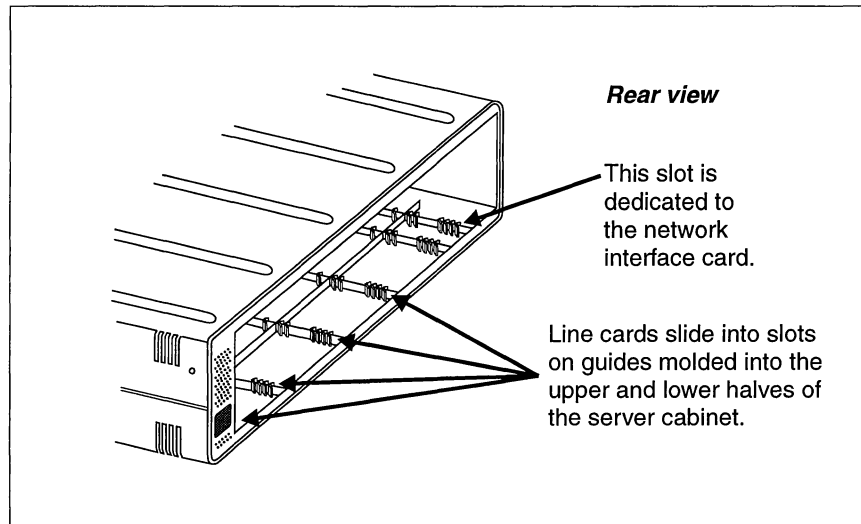


Figure 1-6. Communications Server line card slots

Line cards are inserted into slots in the server chassis. See figure 1-6. Of the five slots, one is dedicated to the ENIC, and the remaining four slots accept any of the available line cards listed below:

Enhanced Network Interface Card (ENIC)—The foundation of the Communications Server is a software-upgradable ENIC, which provides protocol administration and LAN attachment. ENICs can support up to 128 users and come cable-ready for standard AUI thickwire and BNC thinwire connections, with optional support for 10BaseT. ENICs are managed via industry-standard SNMP MIB II and proprietary Penril Datability Networks MIBs. You can select any combination of these protocols according to your network topology:

1. **TCP/IP ENIC** for UNIX host connections implements Telnet in full compliance with all Internet Protocols, as well as TN3270, RLOGIN, Dynamic SLIP, Compressed SLIP, PPP, BOOTP, TFTP, SNMP and enhanced TACACS security.
2. **LAT ENIC** for DEC host connections offers DEC users 100% LAT-compatible operation with host initiated requests, network printer support, network management support and reverse LAT.
3. **TN3270** for connection to IBM hosts. LAN-based or directly connected asynchronous terminals can “telnet” to IBM hosts.

- **Eight-port RS232 line card, 16-port RS423 line card and 32-port RS423 line card**—These line cards provide a fast and easy way to connect eight, 16 or 32 terminals, PCs, printers and modems to the Communications Server. Each supports both V.32bis/V.42bis modems, with the 8-port card achieving a peak data transfer rate of 57.6 Kbps and the 16- and 32-port cards providing data transfer at 38.4 Kbps. You can combine up to four 32-port cards to connect as many as 128 locally attached terminals to one Communications Server.
- **Network Protocol Translator line card**—The NPT line card supports single-protocol (LAT or TCP/IP) terminal servers that require access to both DEC LAT and UNIX TCP/IP hosts by providing advanced transparent protocol translation. Protocol translation capability enables up to 16 simultaneous LAT-to-TCP/IP and TCP/IP-to-LAT communications sessions so that UNIX users can easily access DEC systems and DEC users can easily access UNIX systems.
- **3270 Cluster Controller Line Card (CCLC)**—Connect both LAT and TCP/IP LAN users to IBM mainframes via a standard synchronous link between the IBM front-end processor and LAN. The 3270 CCLC fully emulates an IBM 3x74 cluster controller, so any user on the network can access IBM host applications transparently—without knowledge of gateways, and with no special software or hardware on the IBM host. The card allows up to 32 simultaneous connections and provides true 3278 Models 2-5 and 3279-LA/3A display emulation, complete with user-defined keyboard mapping and hotkey switches between DEC, UNIX and SNA sessions.
- **X.25 PAD card**—The X.25 PAD card adds X.25 gateway capabilities to the Communications Server with bi-directional communication between any LAT and TCP/IP devices on the LAN and private and public data WANs. With the X.25 PAD card, network users can communicate transparently without knowing X.25 addresses or learning PAD commands. The PAD card supports speeds up to 64 Kbps and 64 virtual circuits. In addition, advanced IP-over-X.25 routing is supported.
- **Multisessions line card**—Installing this line card allows Communications Server users to interface with special “windowing” terminals (such as the DEC VT420) that can maintain communications over two terminal sessions at the same time. The special feature of the windowing terminal is its ability to display two screens at the same time. Pressing a function key allows a user to toggle between terminal session screens. Multisessions operations are largely transparent to the user. That is, after logging into the local server, and starting the multisessions feature, users can connect to or disconnect from network services, as desired. The screen display is the only reminder that two terminal sessions are active.

Obtaining Additional Information

Each line card is supplied with a user's guide that describes installation procedures, configuration settings, and line card operations. Some guides also contain listings of commands and messages. For further information about these line cards, see your line card user's guide.

- *X.25 PAD Card User's Guide*
- *3270 Cluster Controller Line Card User's Guide*
- *Multisession Line Card User's Guide*
- *VSM User's Guide*
- *Serial to Parallel Converter Operations Guide*

Additional server information is provided in the *Communications Server Installation and Service Guide*.

To order additional copies of these publications, contact your service representative.

Communications Server and ENIC specifications are given in table 1-3.

Table 1-3. Communications Server and ENIC Specifications

ENIC	
Supported packet types	Ethernet
Routed protocols	LAT, IP
LAT specifications	100% LAT compatible, all management software including NCP and TSM, network printer support, reverse LAT support. Host Initiated Request support.
TCP/IP specifications	Full IP: Compatible with MIL-STD-1777, Complies with RFCs 791, 1122 Full TCP: Compatible with MIL-STD-1778, Complies with RFCs 793, 1122 Telnet: Compatible with MIL-STD-1782, Complies with RFCs 854-861, 1123 ARP: Complies with RFC 826 SLIP: Complies with RFCs 1055, 1144 PPP: Complies with RFCs 1331, 1332 RIP: RFC 1058 TFTP: Complies with RFC 783 RLOGIN, BOOTP: Complies with RFCs 783, 906, 951, 1048, 1084 SNMP: Complies with RFCs 1089, 1098 ICMP: Complies with RFCs 792, 1122 Domain Name System: Complies with RFC 920 Multiple IP Addresses: Allows direction of incoming connections to a group of ports
Processor	20 MHz Intel 80186
Ethernet coprocessor	Intel 82856
Memory	2 Mb dynamic RAM, 128 K non-volatile static RAM, 1 Mb ROM
Memory card	JEIDA/PCMCIA 1.0 standard memory card interface, supports <i>SmartRAMCARD</i>
Network interface	802.3/Ethernet standard thinwire BNC or AUI ports, auto or software-selectable interface.

BASE CHASSIS	
Slot capacity	One ENIC, four line cards
Front panel	Full diagnostic and configuration functions Four line by 40-character backlit LCD display Alphanumeric keypad
Power	90-132/180-264 VAC 47-63 Hz, automatic self-sensing 75W, 225 BTU/hr (typical) 150 W, 510 BTU/hr (max) 1.3 A @ 110 VAC 0.65 A @ 220 VAC
Dimensions	Height: 5¼ inches Width: 17⅝ inches Depth: 17⅞ inches
Weight	10 lb
Environmental	Operating temperature: 0-50o C Humidity: 10-90% non-condensing
Agency approvals	Emissions: FCC part 15, Subpart J Class A VDE 0871 level B Safety: UL478, CSA 22.2, EN 60950, TUV GS Mark

Chapter 2

Configuring the Communications Server

Configuration

In its factory default configuration, the Communications Server supports outgoing connections to network hosts. No changes to the default configuration are required. However, if you will be adding local services, Telnet hosts or links to remote devices, you will need to modify the basic server configuration.

Two Configuration Methods

Server port profiles can be configured two ways:

- Via the **Configurator**
- Via the **parser**

The Configurator is a simple, menu-driven interface you can use to set up SLIP or PPP links between the local server and a target Telnet host. Menu screens prompt you for each of the parameters required to configure a link. Menu screens appear automatically—when you complete one entry, the next menu appears.

The Configurator does not handle all possible server commands and parameters. However, you also can enter commands through the server's parser (that is, at the `Local>` prompt) to create links and perform other server functions. Using the parser, command keywords, characteristics and variable values are typed on a logical command line and executed. For example:

```
Local>      change server bootserver address 123.45.6.7 [ENTER]
```

For further information about using the parser interface to execute configuration commands, see *Configuration Using the Parser*, later in this chapter.

Before You Begin Configuration

Before the Communications Server can be configured, it must be powered up and initialized, and a *privileged* user must be logged in. Proceed as follows:

1. Install the ENIC and line cards. For complete instructions, see the *Communications Server Installation and Service Guide*.
2. Connect the power cable to the Communications Server. Connect the Ethernet cable to the appropriate receptacle on the ENIC backplate.
3. Connect a user terminal to an available server port in an 8-, 16- or 32-port line card, and power on the terminal.
4. Power on the server to start the server initialization diagnostics.
5. Press **ENTER ENTER** to initialize the server port. The Enter Username> prompt is displayed.
6. Log into the server by typing your username and pressing **ENTER**.
7. Your port must have privileged status to use the configurator. Set privileged status with the following commands:

```
Local>      set privileged [ENTER]
```

```
Password>  {password} [ENTER]
```

Note: The default password is "system."

The server is now ready to be configured.

Configuration Using the Configurator

The configurator is a menu-driven interface you use to create both PPP links and SLIP links between the Communications Server and remote hosts. During the configuration process, you are prompted for each of the required parameters and you type your entries in the highlighted field. Some fields are empty and must be populated by you. Other fields are already populated with default settings that you can either leave alone or change to other values. New screens appear one after another until the configuration is complete.

Note: Server installation is described in the *Communications Server Installation and Service Guide*.

Configurator Screen Layout

Each configuration screen is unique, containing different fields, but all screens share the basic layout shown in figure 2-1.

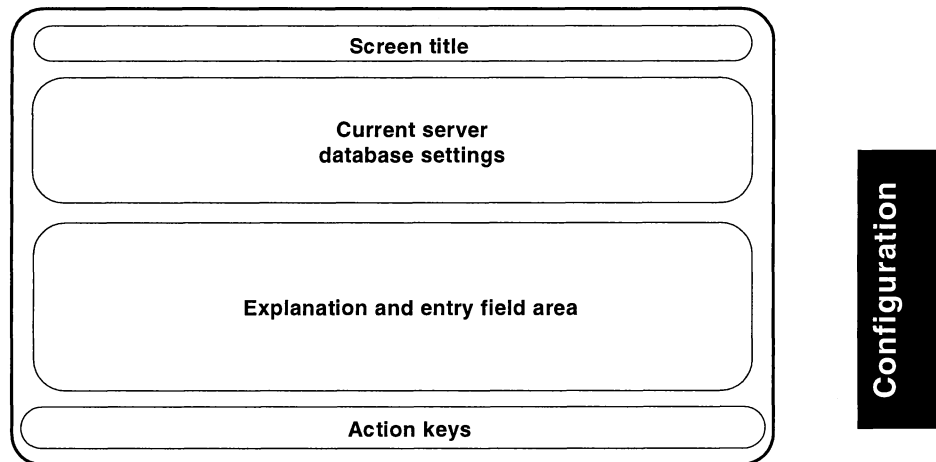


Figure 2-1. Configurator screen layout

The screen areas shown in figure 2-1 are explained below.

- **Screen title**—Summarizes the purpose of the database settings that are displayed in the current screen—for example, Add Link Setup.
- **Current server database settings**—This is the image of the database as it currently exists in the server. The upper-leftmost field is automatically highlighted when you open the screen. To step through the fields, press **ENTER**. Default values and current settings can be changed or overwritten. You populate the database parameter fields with numerical values or character strings. Entries are made at a Config> prompt in the Explanation and Entry Field area.
- **Explanation and entry field area**—This area of the configurator screen has two functions: (1) It offers help text, which explains the purpose of each field in the Current Server Data Base Settings area (toggle the help text on or off by pressing **CTRL- Z**), and (2) it includes a data entry prompt. Entries are either random alphanumeric strings, such as usernames or passwords; or a choice between several preset strings, such as enabled or disabled. Where a random string of characters is required, enter an alphanumeric string beside the entry prompt. Where a choice is indicated, press **SPACE** to toggle through the choices. Press **ENTER** to download your entry.

- **Action keys**—Press the **CTRL** key together with **Z**, **X**, **V** or **L** to execute the actions indicated below:

Press **CTRL-Z** to turn help on or off.

Press **CTRL-X** to exit the configurator.

Press **CTRL-V** to return to the previous field.

Press **CTRL-L** to abandon your changes.

Starting the Configurator

1. You must have privileged status to use the configurator. Awaken the local port. Execute:

[ENTER] [ENTER]

2. You are prompted with the following:

```
Communications Platform      Version Number
Please type HELP if you need assistance
```

3. At the Enter username> prompt, enter your port username. Execute:

Enter username> {username string} **[ENTER]**

4. At the > prompt, set port privileges. Execute:

Local> **set privileged** **[ENTER]**

5. At the **Password>** prompt, enter the privileged password. Execute:

Password> {password} **[ENTER]**

Note: The default password is “system.”

6. Now your port is in privileged mode.

Local>

If this is the first time the server has been configured, refer to the *First Time Configuration* section. To change an existing configuration, see *Reconfiguration*.

First Time Configuration

If your server has never been configured, you must first assign an IP address. Proceed as follows:

1. Start the configurator. Execute:

```
Local>    config [ENTER]
```

The Required IP Setup screen, as shown in figure 2-2, appears if the server has never been configured.

Server Configuration - Required IP Setup

IP Address:

Enter the IP address of this Server and press [ENTER].

Config>

Each Internet host is assigned a unique 32 bit IP address that is used in all communication with that host. IP addresses are assigned centrally by Network Information Center at SRI International and are known by your network manager. IP addresses appear as four decimal integers separated by dots, (i.i.i.i) where each integer gives value of one octet of IP address. i is from 0 to 255.

^Z toggles this help information on or off.

<^Z>-Toggle Help <^X>-Exit <^V>-Previous Field <^L>-Re-enter Screen

Figure 2-2. Required IP Setup screen

2. Obtain the IP address of the local server and enter that address in the highlighted window. Press **ENTER** to download the address.
3. After you press **ENTER**, the following prompt appears:

```
Is the configuration acceptable?
Select <Y> to confirm the configuration or <N> to reenter.
```

If you select **Y**, the following message appears:

```
Executing commands:
CHANGE ADDRESS {local IP address}
Press any key to continue
```

This message contains the command that the server is executing. If you select **N**, the following appears:

```
Configuration aborted
Press any key to reenter screen
```

Press any key and the Required IP Setup screen reappears. Note that the IP address you entered still appears on the screen and remains there until you enter a new IP address. Repeat the entry procedure until you have entered the correct IP address. When you have entered the IP address, go to step 4.

4. After the commands are sent, press any key to display the Configuration main menu, figure 2-3. This screen permits you to either add a new link from the server to a remote host or to exit the configurator and end the server configuration process.

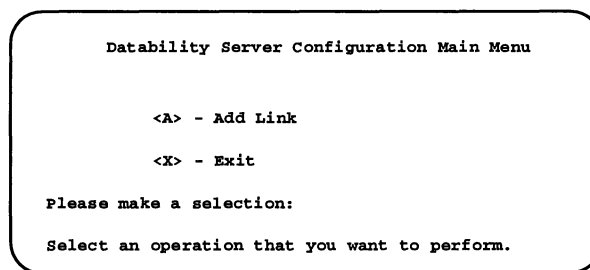


Figure 2-3. Configuration main menu

5. Press **A** to start the link configuration process and to display the Add Link Setup screen (explained following the *Reconfiguration* section).

Reconfiguration

If the server already has an IP address, the first screen you see when you execute the **config** command is the configuration main menu (figure 2-3). This screen permits you to either add a new link from the server to a remote host or to exit the configurator and end the server configuration process.

Adding Links to the Server

Figure 2-4 shows the Add Link Setup screen. The next two sections show how to use the Link Protocol field to specify the protocol to be used and the Port List field to enter the numbers of the server ports to be assigned to the link.

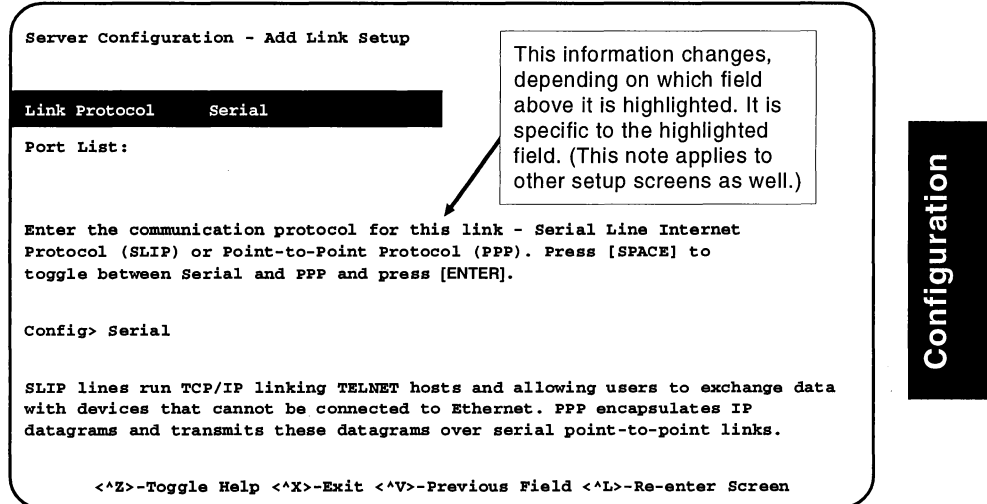


Figure 2-4. Add Link Setup screen

Link Protocol

In the Link Protocol field, select the communications link protocol: Serial (Serial Line Internet Protocol) or PPP (Point-to-Point Protocol). SLIP lines run TCP/IP and link Telnet hosts, allowing users to exchange data with devices that cannot be directly connected to the Ethernet. PPP lines encapsulate and transmit IP datagrams over serial point-to-point links.

1. Toggle to the desired protocol (Serial or PPP) and press **ENTER**.
2. The highlight moves to the Port List field.

Port List

All data exchanges across the serial link occur between a port on the local server and a target port on the remote host. Using the Port List field, you can assign one or more ports to the link. Parameters of server ports assigned to the link are configured accordingly.

1. Enter the numbers of the server ports assigned to this link. Valid port numbers are from 1 to 128. Enter either a series (1,3,5) or a range (1-5) of port numbers and press **ENTER**.

2. The prompt `Is the configuration acceptable?` appears. If the configuration up to this point is okay, press **Y**. If you want to reenter a value, press **N**.
- 3a. If you pressed **Y**, and you specified `Serial` for your Link Protocol, the next screen to appear is the first Serial Line Internet Protocol (SLIP) Setup screen (which is explained in the next section).
- 3b. If you pressed **Y**, and you specified `PPP` for your Link Protocol, the next screen to appear is the Point-to-Point Protocol (PPP) Setup screen (which is explained in a subsequent section).
- 3c. If you pressed **N**, you will see: `Configuration aborted. Press any key to reenter screen.` After pressing a key, the highlight moves back to the Link Protocol field. Re-enter either parameter and press **Y**.

If you selected `Serial` as the link protocol, see the subsequent section titled *Serial Line Interface Protocol (SLIP) Setup*. If you selected `PPP` as the link protocol, see the subsequent sections titled *Point-to-Point Protocol (PPP) IP Setup* and *Point-to-Point Protocol (PPP) Setup*.

Serial Line Interface Protocol (SLIP) Setup

If you chose Serial as the link protocol in the Add Link Setup screen, the Serial Line Internet Protocol (SLIP) Setup screen appears (figure 2-5). The Serial Connection Speed field is highlighted. The Port Number indicated is taken from the list of ports assigned to the link and appears automatically. This screen reappears for each port until all ports assigned to the link have been configured.

Server Configuration - Serial Line Internet Protocol (SLIP) Setup

Port Number: 1

Serial Connection Speed: 9600

Link Destination IP address:

Select the baud rate at which communications on this link will occur between the local server port and the target host. Press [SPACE] to toggle through the choices. Press [ENTER] to select a speed.

Config> 9600

Serial Connection Speed value is determined by comparing maximum allowable baud rate at target device with maximum allowable baud rate at local server. Set connection speed to the lower of the two maximum baud rates.

<^Z>-Toggle Help <^X>-Exit <^V>-Previous Field <^L>-Re-enter Screen

Configuration

Figure 2-5. SLIP Setup screen

The next two sections explain the Serial Connection Speed and Link Destination IP Address fields.

Serial Connection Speed

1. Press **SPACE** to toggle through the available serial port speeds (110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400 or 57600). The maximum allowable port speed depends upon the line card in which the target port resides, as indicated in the following table:

Line card	Max. port speed
8 Port Line Card	57.6 Kbps
16 Port Line Card	38.4 Kbps
32 Port Line Card	38.4 Kbps

Note: You cannot specify the port through which you currently are logged in.

2. Press **ENTER** to select a speed. The default is 9600 baud.
3. The highlight moves to the Link Destination IP Address field and a new screen appears.

Link Destination IP Address

The link destination IP address is the IP address of the remote host with which the local server will be communicating via the link.

Each Internet host is assigned a unique 32-bit IP address used in all communication with that host. IP addresses are assigned centrally by the Network Information Center and are known by your network manager. You can reach the NIC by telephone at 1-800-365-2578, or by mail at: DDN Network Information Center, 14200 Park Meadow Drive, Suite 200, Chantilly, Va. 22021, U.S.A.

1. Enter the digits of the IP address of the remote host with which the local server will be communicating.

Note: Press the backspace key to erase incorrect digits.

2. Press **ENTER** to download the IP address to the server.
3. A prompt appears:

Is the configuration acceptable?

- 3a. If the displayed settings are okay, press **Y**. A new display shows the current configuration settings in parser format.
- 3b. If the configuration settings are *not* correct and you want to change something, press **N**. The following message appears:

Configuration aborted. Press any key to reenter screen.

Press a key and the highlight returns to the Link Protocol field.
Re-enter either parameter and press **Y**.

4. If you selected multiple server ports in the Add Link Setup screen, you are prompted with a SLIP setup screen for each of the remaining ports. Repeat the procedure until you have completed configuring all ports assigned to the SLIP link.

When you have configured all ports and have answered **Y** to the Configuration Acceptable? prompt, the screen returns to the main menu, where you are prompted to either add another link or to exit the configurator.

If you choose to add another link, that link can be either a SLIP link or a PPP link. The following pages explain how to configure a PPP link.

Point-to-Point Protocol (PPP) IP Setup

If you chose PPP as the link protocol in the Add Link Setup screen, the Point-to-Point Protocol (PPP) IP Setup screen appears (figure 2-6). The Serial Connection Speed field is highlighted. The Port Number indicated is taken from the list of ports assigned to the link and appears automatically. This screen reappears for each port until all ports assigned to the link have been configured.

Server Configuration - Point-to-Point Protocol (PPP) IP Setup

Port Number: 1

Serial Connection Speed: 9600

Link Destination IP Address:

Select the baud rate at which communications on this link will occur between the local server port and the target host. Press [SPACE] to toggle through the choices. Press [ENTER] to select a speed.

Config> 9600

Serial Connection Speed value is determined by comparing maximum allowable baud rate at target device with maximum allowable baud rate at local server. Set connection speed to the lower of the two maximum baud rates.

<^Z>-Toggle Help <^X>-Exit <^V>-Previous Field <^L>-Re-enter Screen

Configuration

Figure 2-6. PPP IP Setup screen

The following sections explain the Serial Connection Speed and Link Destination IP Address fields.

Serial Connection Speed

1. Press **SPACE** to toggle through the available serial port speeds (110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400 or 57600). The maximum allowable port speed depends upon the line card in which the target port resides, as indicated in the following table:

Line card	Max. port speed
8 Port Line Card	57.6 Kbps
16 Port Line Card	38.4 Kbps
32 Port Line Card	38.4 Kbps

2. Press **ENTER** to select a speed. The default is 9600 baud.
3. The highlight moves to the Link Destination IP Address field and a new screen appears.

Link Destination IP Address

The link destination IP address is the IP address of the remote host with which the local server will be communicating via the link.

Each Internet host is assigned a unique 32-bit IP address used in all communication with that host. IP addresses are assigned centrally by the Network Information Center and are known by your network manager. You can reach the NIC by telephone at 1-800-365-2578, or by mail at: DDN Network Information Center, 14200 Park Meadow Drive, Suite 200, Chantilly, Va. 22021, U.S.A.

1. Enter the digits of the IP address of the remote host with which the local server will be communicating.

Note: Press the backspace key to erase incorrect digits.

2. Press **ENTER** to download the IP address to the server.

3. The following prompt appears:

Is the configuration acceptable?

- 3a. If the displayed settings are okay, press **Y**. A new display shows the Point-to-Point Protocol (PPP) Setup screen. (This screen is explained in the following sections.)
- 3b. If the configuration settings are not correct and you want to change something, press **N**. Configuration aborted. Press any key to reenter screen appears. Press a key and the highlight returns to the Link Protocol field. Re-enter either parameter and press **Y**.
4. If you selected multiple server ports in the Add Link Setup screen, you are prompted with a PPP setup screen for each of the remaining ports. Repeat the procedure until you have completed configuring all ports assigned to the PPP link.

When you have configured all ports and have answered **Y** to the Configuration Acceptable? prompt, the screen returns to the main menu, where you are prompted to either add another link or to exit the configurator.

If you choose to add another link, that link can be either a SLIP link or a PPP link.

Point-to-Point Protocol (PPP) Setup

When you accept the IP address configuration displayed on the Point-to-Point Protocol (PPP) IP Setup screen, the server displays the Point-to-Point Protocol (PPP) Setup screen (figure 2-7).

Server Configuration - Point-to-Point Protocol (PPP) Setup

Port Number: 1

Authentication Protocol:	TABLE	Magic Number:	DISABLE
Quality monitor:	DISABLE	Proto Compress:	DISABLE
Address Compress:	DISABLE	FCS:	16
State:	CLOSED	Link Authentication:	ANY
Maximum Receive Unit Min:	200	Maximum Receive Unit Max:	1500
Maximum Receive Unit Actual:	1500	Async Map:	ESCAPE
Async Map Mask:	0-31		

Determine how the local server will authenticate PPP usernames and passwords. Choices are: TABLE and TACACS. Press [SPACE] to toggle between the choices and press [ENTER] to select.

Config> TABLE

Upon receiving a PAP or CHAP packet, the local server authenticates a connection request either according to username and password data stored in its local authentication table or according to the TACACS method.

<^Z>-Toggle Help <^X>-Exit <^V>-Previous Field <^L>-Re-enter Screen

Configuration

Figure 2-7. PPP Setup screen

The fields in this screen are explained in the following sections.

Authentication Protocol

Follow these steps to select an authentication protocol:

1. Press **SPACE** to toggle through the available authentication types: TACACS or TABLE. The default is TABLE. The Authentication Protocol must be compatible with the Link Authentication, as indicated in the following table:

Authentication Compatibility:		
Link authentication	Authentication protocol	
	TABLE	TACACS
PAP	YES	YES
CHAP	YES	NO
ANY	YES	NO
NONE	NO	NO
YES Compatible; NO Not compatible.		

2. Press **ENTER** to select an authentication type.
3. The highlight moves to the Magic Number field on the associated screen. (This field is described in the following section.)

Magic Number

The Magic Number parameter activates an option that uses a randomly selected number to detect looped-back links and monitor link quality.

1. Enable or disable the Magic Number parameter across this PPP link. Press **SPACE** to toggle between the choices. The default is DISABLE.
2. Press **ENTER** to download your selection.
3. The highlight moves to the Quality Monitor field on the associated screen (described in the next section).

Quality Monitor

Quality Monitor controls the transmission of data link quality report packets. If enabled, the Quality Monitor can determine if and how often data packets are being dropped on the link between the source port and the target host.

1. Press **SPACE** to toggle between ENABLE and DISABLE. The default is DISABLE.
2. Press **ENTER** to select your choice.
3. The highlight moves to the Proto Compress field on the associated screen (described in the next section).

Protocol Compression

The Proto Compress field enables or disables protocol compression. Enable protocol compression to inform the remote host (during options negotiation) that the local server will accept PPP frames containing a compressed data link layer protocol field. This feature is useful in lower speed links.

1. Press **SPACE** to toggle from ENABLE to DISABLE. The default is DISABLE.
2. Press **ENTER** to select your choice.
3. The highlight moves to the Address Compression field on the associated screen.

Address Compression

The Address Compress field enables or disables address compression. Enable address compression to reduce the length of both the Address and Control fields to null characters within the header of each outgoing PPP packet. The Enabled flag tells a remote host that the local server can accept compressed frames.

1. Press **SPACE** to toggle from ENABLE to DISABLE. The default is DISABLE.
2. Press **ENTER** to select your choice.
3. The highlight moves to the FCS field on the associated screen.

Frame Checking Sequence (FCS)

Use the FCS field to activate either a 16-bit or 32-bit Frame Checking Sequence. The sequence is used to calculate a checksum value that verifies data accuracy in each incoming PPP packet.

1. Press **SPACE** to toggle between 16-bit and 32-bit Frame Checking Sequences. The default is 16-bit FCS. Note that 32-bit Frame Checking is not supported in this software release.
2. Press **ENTER** to select your choice.
3. The highlight moves to the State field on the associated screen.

State

The State field activates or deactivates the PPP link. State OPEN activates the specified PPP link, allowing you to connect to target hosts. State CLOSED deactivates the link, preventing you from connecting to remote hosts via the PPP link.

1. Press **SPACE** to toggle between OPEN and CLOSED. The default is CLOSED.
2. Press **ENTER** to download your selection.
3. The highlight moves to the Link Authentication field on the associated screen.

Link Authentication

Link authentication verifies users before they exchange packets with hosts via the PPP link. The ANY setting adapts the local server to any authentication protocol. If ANY fails, the server defaults to Challenge Handshake Authentication Protocol (CHAP). If CHAP fails, the server next defaults to Password Authentication Protocol (PAP). NONE disables all authentication.

1. Press **SPACE** to toggle through ANY, CHAP, PAP and NONE. The default is ANY authentication protocol. The Authentication Protocol must be compatible with the Link Authentication, as indicated in the following table:

Authentication Compatibility:		
Link authentication	Authentication protocol	
	TABLE	TACACS
PAP	YES	YES
CHAP	YES	NO
ANY	YES	NO
NONE	NO	NO
YES Compatible; NO Not compatible.		

2. Press **ENTER** to download your selection.
3. The highlight moves to the Maximum Receive Unit Min field on the associated screen.

Maximum Receive Unit MIN

The Maximum Receive Unit Min (MRUMIN) field is used to specify the largest packet that can be accepted in one direction (toward the local server port) across the PPP link. The parameter helps the remote host assemble the smallest acceptable frames and dispatch them to the local server.

1. Enter a value between 200 and 1500. The default is 200.
2. Press **ENTER** to download your selected value.
3. The highlight moves to the Maximum Receive Unit Max field on the associated screen.

Maximum Receive Unit MAX

The Maximum Receive Unit Max (MRUMAX) field is used to specify the *upper* limit of the largest packet that can be accepted in one direction (toward the local server port) across the PPP link.

The parameter helps the remote host assemble the largest acceptable frames and dispatch them to the local server.

1. Select a value between 200 and 1500. The default is 1500.
2. Press **ENTER** to download your selected value.
3. The highlight moves to the Maximum Receive Unit Actual field on the associated screen.

Maximum Receive Unit ACTUAL

The Maximum Receive Unit Actual (MRUACTUAL) field is used to specify the size of the largest packet sent in one direction (toward the local server port) across the PPP link. The parameter is populated during options negotiation between the local server and the remote host.

If the MRU maximum receive value is set to less than 1500, the local server still *must* be able to receive 1500 octet frames in the event that link synchronization is lost. Therefore, although you can edit this value through the configurator, it should not be changed from its default setting.

1. Do not change this parameter from its default setting of 1500.
2. Press **ENTER** to bypass the field.
3. The highlight moves to the Async Map field on the associated screen.

Async Map

The Async Map parameter instructs the server to either clear or not clear (escape) ASCII values listed in the Async Map Mask during data transmission via the PPP link.

By default, PPP maps all control characters into an appropriate two-character sequence. However, it is rarely necessary to map all control characters and it may be unnecessary to map any characters. A PPP implementation uses the Async Map parameter to tell the remote host which control characters must remain mapped and which control characters need not remain mapped when the remote host sends them.

1. Press **SPACE** to toggle between ESCAPE and CLEAR. The default is ESCAPE.
2. Press **ENTER** to download your selection.
3. The highlight moves to the Async Map Mask field on the associated screen.

Async Map Mask

The Async Map Mask parameter instructs the server to apply the listed ASCII values to the Async Map parameter during data transmission via the PPP link.

1. Select numerals between 0 and 31 corresponding to ASCII values. The default is all characters from 0 - 31. Write the numerals either as a range (6) or as a series (2,4,8) of numbers.
2. Press **ENTER** to download your selection.
3. Since this is the final field on the Point-to-Point Protocol Setup screen, the following prompt appears:

Is the configuration acceptable?

- 3a. If the displayed settings are okay, press **Y**. A new display (figure 2-8) shows the current configuration settings in parser format. These are the configuration commands that the server will execute when you leave this screen. The values in this screen are populated either *manually* by the user or *automatically* by the server. For example, the interface number and the Ethernet address are automatically provided by the server, while other values, such as port number and interface type, are specified by the user during server configuration.
- 3b. If the displayed settings are not correct and you want to change something, press **N**. Configuration aborted. Press any key to reenter screen appears. Press a key and the highlight returns to the Authentication Protocol field. You can begin changing values in screen fields as necessary.

Executing commands:

CHANGE INTERFACE -dummy- TYPE PPP PORT 1
CH PORT 1 ACCESS DYNAMIC AUTOBAUD DIS FLOW CONTROL DIS SPEED 9600
CH ROUTE 1 DEST 1.1.1.1 GATE 1.1.1.1 INTERFACE -dummy- TYPE HOST
CH ARP 1.1.1.1 ETHERNET 11-22-344-55-66 PROXY ENA MASK 255.255.255.255
CH PPP LINK 1 MAG DISABLE QUA DISABLE PRO DISABLE ADD DISABLE FCS 16 STA CLOSED
CH PPP LINK 1 AUTHENT ANY AUTH_PR TACACS MRUMIN 200 MRUMAX 1500 MRUACT 1500
CH PPP LINK 1 ASYNC_MAP 0-31 ESCAPE
Press any key to continue

Figure 2-8. Configuration settings in parser format

- If you specified multiple server ports in the Port List field of the Add Link Setup screen, you are prompted with a PPP setup screen for each of the remaining ports. Repeat the procedure until you have completed configuring all ports assigned to the PPP link.

When you have configured all ports and have answered **Y** to the Configuration Acceptable? prompt, the screen returns to the

Server Configuration Main Menu, where you are prompted to either add another link or to exit the configurator.

- Press **A** to define another link and display the Add Link Setup screen (previously described).
- Press **X** to exit from the configurator and return to the Local> prompt.

If you choose to add another link, that link can be either a SLIP link or a PPP link.

Configuration Using the Parser

You can configure the Communications Server to support devices such as printers and modems, and to interface with remote hosts and devices via links. Each of these operations requires some additional set up or modification of the server's default configuration. The modifications are easily accomplished either via the configurator (primarily to establish links) or via the parser.

Unlike the configurator, which guides you through a series of data entry fields in successive screens, the parser (figure 2-8) prompts you for a command string, which you enter and execute from the keyboard of your terminal. You determine the content of the command string; that is, you indicate the port and server characteristics and variable values. You choose these commands from listings indicated in the on-line Help.

The procedures in this section are designed to help you speed through the configuration process. To use this section, locate the procedure that most closely matches the configuration you desire and follow the steps outlined in that procedure. Pertinent examples are provided along the way to illustrate how the complete command line should look when executed. Of course, the procedures can be modified to fit your exact requirements. Following is a list of configuration topics covered in this section:

- IPX Netware Print Server Configuration
- LP Printing
- LAT Service Configuration
- PPP Interface Configuration
- RIP Interface Configuration
- SLIP Interface Configuration
- Telnet Service Configuration
- TN3270 Configuration
- Network Protocol Translation.

IPX Netware—Print Server Configuration

Local area networks often support the host processors, user terminals and output devices of multiple vendors such as IBM, DEC, Sun and Novell. Consequently, the TCP/IP, LAT and IPX protocols that transfer data between devices often run concurrently on the local Ethernet. At the same time, users are adding more applications to help them do their work, and they are installing more printers to produce hard copy output. In such an environment, it is not uncommon to find stand-alone printers segmented by application so that groups of printers are dedicated to either DEC VAX-based or Novell-based applications.

Dedicating printers to specific applications can be wasteful in terms of both the costs of purchasing and maintaining redundant hardware, and the complexity of administering many isolated printing systems.

A more effective solution is to connect the isolated printers to a network print server that can manage the output files of many separate applications and direct those files to the proper printer (figure 2-9). The Communications Server with ENIC^{Plus} provides this important solution to corporate printing needs.

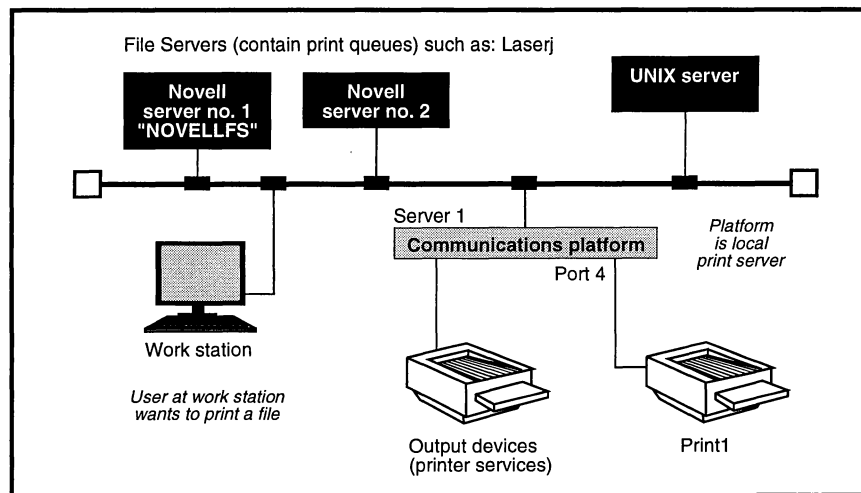


Figure 2-9. ENIC^{Plus} used to direct print jobs

ENIC^{Plus} includes LAT, TCP/IP and IPX printing support. With the addition of IPX printing support, the Communications Server with ENIC^{Plus} becomes a high-performance, multi-protocol print server that prints output files from DEC VAX-based, UNIX-based and Novell network-based devices. Using IPX printing support, printers can be shared across a network, so there is no need to dedicate a printer to a specific application.

Figure 2-9 illustrates a network configuration in which a user at a workstation sends a print file to a print queue (Laserj) on Novell file server no. 1 (Novellfs), which eventually downloads the file to Print1, an output device attached to port 4 of Server1, the ENIC^{Plus} Server.

IPX Protocol

Internetwork Packet Exchange (IPX) protocol routes packets and makes a best effort to deliver them. Structurally, modules of IPX protocol software are stacked vertically into layers. Each layer takes responsibility for handling one part of the data transfer process. For example, incoming packets are passed upward from a data link protocol layer to the IPX protocol layer which, in turn, passes packets upward to additional protocol layers. See figure 2-10.

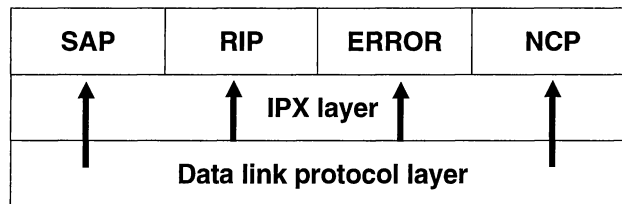


Figure 2-10. Structure of IPX protocol

The IPX protocol layer passes packets upward to other protocols that operate on top of IPX protocol. These are:

- **Service Announcement Protocol (SAP)**—Every 60 seconds, file and print servers announce what services they provide. The local Communications Server listens to these announcements and collects address information about network print server nodes to dynamically update its database.
- **Routing Information Protocol (RIP)**—RIP gathers all of the routes in the network.
- **Error Protocol**—This part of the protocol handles error and control messages.
- **Network Core Protocol (NCP)**—Novell uses this protocol to manage all aspects of client-server data transmissions, such as handling queue requests and servicing the queues. The IPX printing implementation on Plus-model Communications Servers supports the following four data link packet types: 802.3, Ethernet II, 802.2 and 802.3 with SNAP headers. When you configure the server for IPX printing support, you select one of these data link types. By default, the data link type is 802.3; however, your network may be different. Consult your network manager for the data link protocol in use on your network.

IPX Printing Via ENIC^{Plus}

When a NetWare user at a workstation sends a print job to an output device such as a laser printer, that file is temporarily stored in a print queue on the Novell file server. At regular intervals during operation, the ENIC^{Plus} server polls the print queue in each of the Novell file servers to check for files that are waiting to print. If the server detects a queued print job, it transfers the queued file to its own queue and starts the printing process.

Configuring for IPX Printing

There are two parts to the IPX printing configuration procedure:

1. Configuring the local server to operate as a network print server using the IPX NetWare protocol.
2. Configuring the NetWare file server to recognize the local server.

Note: Configuration is done via the PCONSOLE utility on the file server. It is assumed you are familiar with the PCONSOLE utility.

Configuration

Configuring the Local Server

To configure the local server to operate as a network print server using the IPX NetWare protocol, proceed as follows:

1. Set your port to privileged operation. Execute:

```
Local>      set privileged [ENTER]
Password>   {privileged password} [ENTER]
```

2. Assign a name to the local server. Use this name when you log into the local print server to check for jobs in the queue. This name is the same as the local print server name. Execute:

```
Local>      change server name {print server name} [ENTER]
```

where:

PRINT SERVER NAME—Is a name assigned to the local ENIC^{Plus} server acting as a print server.

For example, to assign the name SERVER1, execute:

```
Local>      change server name server1 [ENTER]
```

3. Assign a local server port to the target printer. Execute:

```
Local> change port {port no.} access dynamic autobaud disabled
io_flush disabled speed {port speed} [ENTER]
```

where:

PORT NUMBER—Local server port assigned to the printer.

PORT SPEED—This speed must be compatible with the data transmission speed supported by the local printer.

For example, to assign the target printer to port four with a baud rate of 9600, execute:

```
Local> change port 4 access dynamic autobaud disabled io_flush
disabled speed 9600 [ENTER]
```

4. Select the type of data link network in use on your local LAN. The interface must be the Ethernet interface. By default, the server's Ethernet interface is zero (0) and the server's IPX data link network setting is 802.3. If the default settings have not been changed, you can omit this step. If the default settings have been changed and you want to restore the default values, or if you want to change the default settings to new values, execute:

```
Local> change interface 0 ipx enable ipx_datalink {data link
type} [ENTER]
```

where:

INTERFACE 0—The interface must be the Ethernet interface, which (in the ENIC^{Plus} server) is zero (0).

IPX ENABLE—Activates the NetWare IPX protocol at the server's Ethernet interface.

DATA LINK TYPE—See the table below:

Entry	Data Link Type
802_3	802_3 (default setting)
ethernet_II	Ethernet II
802_3_snap	802.3 with Snap headers
802_2	802.2

For example, to select the 802.3 data link, execute:

```
Local> change interface 0 ipx enable ipx_datalink 802_3 [ENTER]
```

5. Designate one or more Novell file servers that will queue print jobs from NetWare users. These are the Novell file servers to which the print server will connect. Execute:

```
Local> change netware server {Novell file server name} [ENTER]
```

For example, to designate *Novellfs* as a Novell file server, execute:

```
Local> change netware server novellfs [ENTER]
```

6. Create a local service pointing to the local printer. Assign (to this local service), the number of the local server port connected to the printer. Enable the netware protocol for that service. Execute:

```
Local> change service {printer service} port {port no.}
       netware enabled [ENTER]
```

where:

LOCAL PRINTER SERVICE—Is an alphanumeric character string identifying the local printer service. This local service name will also be used as a NetWare print queue name on the Novell side of the configuration. During operations, the local server will poll the NetWare print queue name every 30 seconds to check for the presence of a queued job. If the local server detects a queued print job, it will send that job to the local printer.

For example, to create the local printer service *print1* at server port four (the port to which the printer is connected), execute:

```
Local> change service print1 port 4 netware enabled [ENTER]
```

Configuring the Novell NetWare Host

To process NetWare print jobs, the ENIC^{Plus} server logs into the Novell file server and queries that server for pending print jobs. The ENIC^{Plus} server logs into the file server using its own name, rather than using the name of one of its local printer services. Therefore, in the procedure outlined below, the ENIC^{Plus} server name is entered along with the printer service name.

Note: Novell NetWare is available in different versions. Any installation using NetWare probably will be configured with site-dependent variables. It is beyond the scope of this *User's Guide* to document configuration procedures for all locations using NetWare.

For additional information about the Novell configuration parameters, consult your system documentation. There are three parts to the NetWare configuration procedure:

1. Setting the NetWare file server to allow unencrypted passwords
2. Telling the NetWare file server about print resources available on the ENIC^{Plus} server
3. Setting print queues on the NetWare file server to use the Communications Server print resources.

Repeat these steps on each NetWare file server that will print through the ENIC^{Plus} server. To print a job, a NetWare user specifies the file server's queue name. Both the file server and the ENIC^{Plus} server determine which of the server's queues will execute the

print request. Proceed as follows:

1. Log into the Novell file server with supervisor privileges. Execute:

```
F:\Login> login supervisor [ENTER]
```

2. When you are logged in, the screen prompt changes to a System> prompt. From the System> prompt, access the main menu. Execute:

```
F:\System> menu main [ENTER]
```

3. The main menu appears. To configure both the printer server and print queue parameters, and to monitor all operations, execute commands through the main menu windows and sub-windows.
 - Use the *up* and *down* arrow keys to move the highlighted window to a new selection.
 - Press **ESC** to exit a sub-menu window and return to a higher level menu window.

Allowing Unencrypted Passwords

4. You will need to tell the file server to allow unencrypted passwords because the ENIC^{Plus} server will *not* use encrypted passwords. This feature was added to NetWare 386 as a security option. By default, password encryption is required, so if you do not change the parameter setting, the ENIC^{Plus} server will not be able to log into the file server and accept print jobs.

To tell the file server to allow unencrypted passwords, proceed as follows:

5. Select System Configuration and press **ENTER**.
6. The Available Topics menu appears. Select Supervisor Options and press **ENTER**.

7. On the Supervisor Options menu, select Edit System Autoexec File and press **ENTER**.
8. The screen changes to show the current AUTOEXEC.NCF file. Add this command line anywhere in the file:

```
set allow unencrypted passwords = on
```
9. Press **ESCAPE** to exit the file editor.
10. Answer yes (**Y**) to the Save Changes? menu.
11. Press **ESCAPE** again to exit the System Configuration utility and return to the main menu.

Configuring Print Resources

12. At this point, you must tell the file server what print resources are on the network. Select Print Queue Management and press **ENTER**.
13. The Available Options menu appears. Select Print Server Information and press **ENTER**.
14. Print Server Information displays a list of the currently configured print servers at this file server. For example:
15. You can edit this list by adding or deleting entries. Press **INSERT** to add a new entry and the New Print Server Name window appears. Type the new print server entry in the allocated space.

```
New Print Server Name:
```

For example, add the print server named *SERVER1*.

```
New Print Server Name: SERVER1
```

16. Press **INSERT** again to add the name of the new print service offered at the new print server. Note that the print service is entered in the same New Print Server Name window. For example, add the printer service named *PRINT1*.

```
New Print Server Name: PRINT1
```

17. Press **ESC** to return to the Available Options menu.

Setting Print Queues

18. At this point, you must create a print queue on the file server and associate that queue with the ENIC^{Plus} server services you just created. Select the Print Queue Information screen and press **ENTER**.

Available Options
Change Current File Server
Print Queue Information
Print Server Information

19. A new screen appears, showing a list of active print queues in the file server. Press **INSERT** to open the New Print Queue Name window.

New Print Queue Name:

The new print queue name, which is used by NetWare users, need not match either of the print server names entered above. Instead, it should be a name that users will easily remember.

20. Type the new print queue name and press **ENTER** to download that name to the file server. For example, to add the name *LASERJ* to the file server, enter the name in the window as follows:

New Print Queue Name: LASERJ

21. Configure the new print queue name to recognize the new print servers you entered above. Begin by highlighting the new print queue name you just entered. Press **ENTER** to configure that queue name. The Print Queue Information screen appears.

Print Queue Information
Current Print Job Entries
Current Queue Status
Currently Attached Servers
Print Queue ID
Queue Operators
Queue Servers
Queue Users

22. Specify which network print servers can print jobs from the new print queue name, entered above. On the Print Queue Information screen, select Queue Servers and press **ENTER**.
23. The Queue Servers screen appears, but the list of print servers is empty because nothing has been selected.

24. Press **INSERT** to produce the Queue Server Candidates screen:

Queue Server Candidate		Queue Servers
Printer1	(Printserver)	
Server1	(Printserver)	
Print1	(Printserver)	
Printer1-1	(Printserver)	

Among the queue server candidates are the print servers you added earlier in this procedure. Both of these print server names must be added to the Queue Servers screen. (In the examples, these were *Server1* and *Print1*.) The server name (*Server1*) is used to log into the file server. The print service name (*Print1*) specifies the target port to which the print jobs will be sent.

25. At the Queue Server Candidates screen, highlight the server name that matches the name of your local server. This is the server you want to attach to the queue. Press **ENTER** to add that name to the list of queue servers.

For example, add SERVER1 to the list of queue servers:

Queue Server Candidate		Queue Servers
Server1	(Printserver)	Server1
Print1	(Printserver)	

The server name now appears on the Queue Servers screen indicating that it is attached to the new print queue name, entered earlier.

26. Return to the Queue Server Candidates screen and highlight the name of the printer service at the local server. Press **ENTER** to add that name to the list of queue servers.

For example, add PRINT1 to the list of queue servers:

Queue Server Candidate		Queue Servers
Server1	(Printserver)	Server1
Print1	(Printserver)	Print1

27. When you have attached the print servers to the new print queue name, repeatedly press **ESCAPE** to leave each of the sub-menus until you are back to the Available Options menu. From this menu you can repeat the procedure to add multiple print queues to the file server.

Available Options	
Change Current File Server	
Print Queue Information	
Print Server Information	

If you are not adding any more print queues, you can repeatedly press **ESCAPE** to exit the PCONSOLE utility.

28. Wait 5 minutes for the Novell host to completely attach these new print queues. Then, you can start printing.

You can shorten the wait if you execute the following command from the local server:

```
Local> set netware server {name of Novell file server} [ENTER]
```

For example, if the Novell file server name is *Novellfs*, execute:

```
Local> set netware server novellfs [ENTER]
```

Printing Via IPX

When you have configured both the local ENIC^{Plus} server and the NetWare side of the connection, you are ready to submit print jobs. Following is a procedure to start a print job. Your Novell system may be configured to operate differently than the system that is documented below. For further information about your system, contact your Novell system manager.

1. Access the Available Options Menu and select Print Queue Information.
2. Select your print queue and press **ENTER**.
3. The Print Queue Information screen is displayed. Select Current Print Job Entries.
4. Select the source directory containing the print file. The default directory appears on the screen.
5. Press **BACKSPACE** to delete the default directory.
6. Press **INSERT** to produce a new screen showing local drives and the default directories. For example:

File Servers/Local Drives	
A:	(Local Drive)
B:	(Local Drive)
C:	(Local Drive)
D:	(Local Drive)
NOVELLFS	Supervisor

7. Select the drive from this listing. Press **ENTER** to produce a new screen showing local directories on that drive. For example:

```
Local Directories
Test1
Test2
June
July
```

8. Select a local directory containing the desired target file. Press **ENTER** to produce a list of sub-directories (if they exist) in that directory.
9. If there are no more sub-directories in that directory, press **ESC** to access the Main Selection Directory screen. For example, if you selected the Test1 directory on the D drive, the screen would look like this:

```
Main Selection Directory
D:\Test1
```

10. Press **ENTER** to generate a list of files in that directory.

```
Available Files
File001
File002
File003
File004
```

11. Select the file you want to print from the list and press **ENTER**.
12. The System Directory screen opens and the PCONSOLE defaults selection is highlighted.

```
System Directory
PCONSOLE Defaults
```

13. Press **ENTER** to display the New Print Job to be Submitted screen and the default settings for the print job. For example, if the file name FILE001 is to be printed, the screen looks like this:

New Print Job to be Submitted			
Print Job:		Filesize:	
Client:	Supervisor		
Description:	File001		
Status:			
User Hold:	No	Job Entry Date:	
Operator Hold:	No	Job Entry Time:	
Service Sequence:			
Number of Copies:	1	Form:	0
File Contents:	Byte Stream	Print Banner:	Yes
Tab size:		Name:	Supervisor
Suppress Form Feed:	No	Banner Name:	File001
Notify When Done:	No		
Target Server:	(Any Server)	Defer Printing:	No
		Target Date:	
		Target Time:	

14. Press **ESC** to submit the print job with the parameter settings shown in the set up screen (above).
15. Select Yes on the Save Changes screen.

Save Changes	
<input checked="" type="radio"/>	Yes
<input type="radio"/>	No

16. The job will begin printing on the target printer.

LP Printing through UNIX

The UNIX LP print service (or *LP spooler*) is a mechanism that lets users send a file to be printed while they continue with other work. The term *spool* is an acronym for Simultaneous Peripheral Output On-Line. *LP* once meant *line printer*, but now includes a variety of printing devices. The LP print service system software:

- Receives the files users want to print
- Filters the files (if necessary) for proper printing
- Schedules the work of one or more printers
- Starts programs that interface with the printers
- Tracks print job status
- Reports printer problems via error messages
- Tracks currently mounted preprinted forms and alerts the user to mount needed forms

Printer sharing functions require cooperative processes in both the host system and the terminal server. In the host, a process typically is associated with the operating system print spooler that executes the shared printer registration and connection protocol. In the terminal server, a single process both listens to and responds to registration requests.

In UNIX hosts, the registration request uses UDP (User Datagram Protocol), and the data are transmitted using the Telnet protocol. However, printing through UNIX has been a challenge to implement properly. Within a UNIX environment, the standard LP print service was designed to either run in a host-to-host environment or to output directly to a local physical device on the UNIX host. For this reason, printing via terminal servers can be a complex issue. The solution is a UNIX printing software program that provides a TTY interface to receive LPD output.

Printing from a Workstation via a UNIX Host

Consider the example of a user at a UNIX workstation who sends a print job to a UNIX host. (See figure 2-11.) The user's print job is spooled to a print queue on the UNIX host either by pseudo terminals or through named pipes. The UNIX printing software program provides a bi-directional raw stream between the UNIX host and the terminal server. No data processing occurs in the program. The UNIX host sends the job to a TCP port on ENIC^{Plus} Server1, which is connected to a printer. (You may have to reconfigure the printer to convert linefeeds into carriage return/linefeed pairs.) Following are procedures to set up this mechanism on the local ENIC^{Plus} server and the UNIX host.

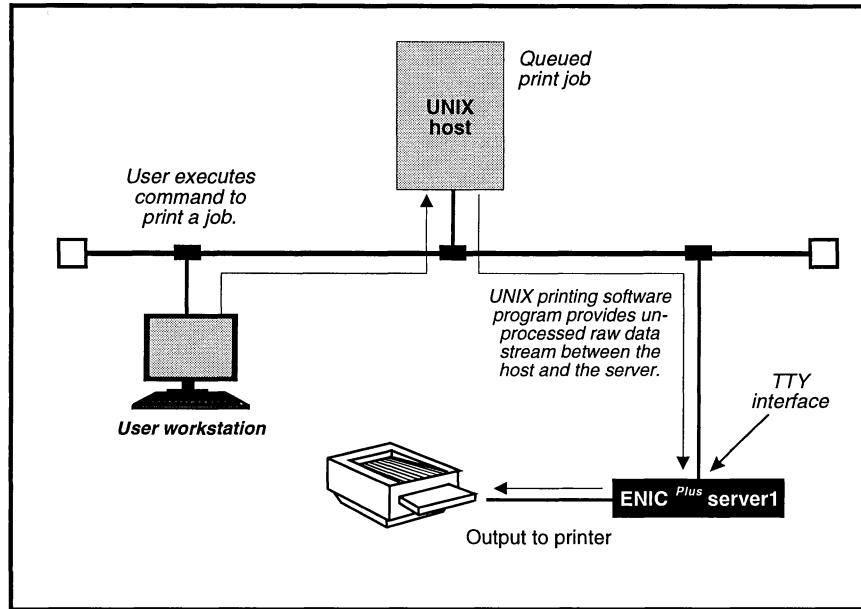


Figure 2-11. UNIX workstation job to printer on ENIC

Configuring the Local Server

1. Assign a server name to the target server that will support the target printer. Execute:

```
Local> change server name {name of server} [ENTER]
```

For example, if the target server is named *Server1*, execute:

```
Local> change server name server1 [ENTER]
```

2. Assign an IP address to the target server. Execute:

```
Local> change address {IP address of server} [ENTER]
```

For example, if the target server's IP address is *1.2.3.4*, execute:

```
Local> change address 1.2.3.4 [ENTER]
```

3. Select a service name that will identify the target printer. Assign a local physical server port to the target printer service. Execute:

```
Local> change service {printer service name} port {server  
physical port number} [ENTER]
```

For example, if the target printer service name is *Uxprint* and the service is assigned to port six, execute:

```
Local> change service uxprint port 6 [ENTER]
```

4. Activate LPR protocol at this service. LPR is used because of difficulties that may be encountered when Telnet sends certain types of files (e.g., postscript files) during a file transfer. In Telnet, certain characters and control functions are interpreted and executed during file transfer. However, when the file is being printed, the interpreter functions are not desired. Therefore, disable the Telnet protocol (and the LAT protocol) at this service to deactivate the interpreting function. Activate the LPR protocol to pass the raw data characters to the target printer without interpreting the characters. Execute:

```
Local> change service {name of service} lpr enabled Telnet
disabled lat disabled [ENTER]
```

For example, if the service name is *Uxprint*, execute:

```
Local> change service uxprint lpr enabled Telnet disabled lat
disabled [ENTER]
```

5. Assign a TCP port number to the service. It is good practice to start using TCP port numbers higher than 2000 or 2100 to avoid using any of the well-known sockets. (2048 is the server console port.) Execute:

```
Local> change service {name of service} tcp port {tcp port
number} [ENTER]
```

For example, if the service name is *Uxprint* and the TCP port number is 7006, execute:

```
Local> change service uxprint tcp port 7006 [ENTER]
```

6. Assign an IP address to the local service. This can be the same as the main IP address of the server or it can be a unique address.

```
Local> change service {name of service} IP {IP address of
service} [ENTER]
```

For example, if the service name is *Uxprint*, and if the service will use the main IP address of the server, 1.2.3.4, execute:

```
Local> change service uxprint ip 1.2.3.4 [ENTER]
```

7. Reboot the ENIC^{Plus} server so the correct service ratings will become effective. Execute:

```
Local> crash [ENTER]
```

8. Proceed to the next section, *UNIX Printing Software Programs*.

UNIX Printing Software Programs

Penril Datability Networks offers two UNIX printing software programs (in C language) and implements them as follows:

- *TCPSEND*, which is used on AT&T-compliant systems
- *TCPXFER*, which is used on BSD-compliant systems

TCPSEND is a filter that acts on the data flowing through it. TCPXFER is a daemon process, which doesn't act on the data flowing through it; it only sends the data to a target.

In operation, the UNIX print spooler invokes the printing software program to direct files to network queues or devices. The server connection queue controls all additional data transfer between the host and the server port offering the shared printer service.

Installing the UNIX Printing Software Programs

You can install the printing software into the target UNIX host in either of the following ways:

1. Dial into the Penril Datability Networks host and FTP them directly to the target host.
2. Penril Datability Networks will copy the UNIX printing software onto a 3½-inch floppy disk and send that disk to you.
 - If your machine has a device that can read these floppies (most IBM RS6000 units have such a device), you can copy these files directly into the target host.
 - If your machine cannot read a 3½-inch floppy disk, you can read the disk into a PC and then copy that file from the PC to the target host via the Ethernet. For further information about these installation options, contact your service representative.

Configuring a UNIX Shared Printer Service

There are two parts to configuring a UNIX shared printer service:

- **Editing and compiling the printing software program**—This is executed at each host.
- **Defining the printer to the UNIX printer sub-system**—This is executed at each printer. In the procedures that follow, it is assumed that you are familiar with the UNIX commands that apply to your system.

Editing and Compiling Printing Software Programs

Different UNIX systems name files differently, locate files differently and set up print queues differently. Because no single configuration adequately describes all of the different types of UNIX systems, the task of describing exactly how to configure the UNIX host to share a printing resource is complicated. This manual contains procedures for each of the following types of UNIX systems:

- AT&T System V release 4.0
- SCO_UNIX systems
- AIX (BSD) UNIX systems

Refer to the section that matches your system and execute the commands as directed. If your UNIX system is not referenced here, you may be able to derive a procedure using these examples as guides. If you cannot proceed, contact your service representative.

AT&T System V release 4.0

1. Log into the UNIX host as root user.
2. From the root directory, create a directory called `tcp`. Execute:
3. Copy files from the TCPSEND printing software program into `/tcp`. Execute:

```
# mkdir tcp [ENTER]

# cp README /tcp [ENTER]
# cp Makefile /tcp [ENTER]
# cp tcpsend.c /tcp [ENTER]
# cp crnl.c /tcp [ENTER]
# cp conff11.model /tcp [ENTER]
```

4. Makefile is divided into sections pertaining to each of the supported types of UNIX systems (e.g., SCO, SUNOS, AIX, etc.). You will need to use a text editor (such as EPSILON or vi) to customize the makefile to your system. Within the file, locate the type of UNIX system you are using, and un-comment the lines of code within that section. Those lines of code will become part of the libraries that are executed.

Locate the heading "For AMIGA" and un-comment these lines:

```
CFLAGS = -DSOCKET
LIBS = -lsocket -lnsl
```

5. Execute the **Make** command to compile the program. The path must point to the libraries. If your login doesn't have the correct path, the **Make** will not find the path of the libraries. Be sure that no errors occur during the compiling process. Execute:

```
# make [ENTER]
```

6. Before you configure a printer that affects the scheduler, stop the scheduler. Execute:

```
# /usr/lib/lpshut [ENTER]
```

7. Change both the ownership of the file and its group ID. This is done both for security reasons and to let all users execute the program. If ownership is not set up correctly, some users may not be authorized to print. Execute:

```
# chown lp tcp send crnl [ENTER]
# chmod 111 tcp send [ENTER]
```

8. Check that the terminal server's IP address is in the `/etc/hosts` file. For example, the display might look like this:

```
1.2.3.4 VISTA test
```

9. Locate the UNIX system directory in which the printer interfaces reside. (Typically, this is the `/usr/lib/lp/model` directory.) Create a file in this directory using the same name as that of the queue. There are two ways to do this:

- Either use your favorite line editor to add the following command line to this file:

```
/usr/lib/lp/model/standard "$@" | /tcp/tcp/crnl | /tcp/tcp send
{server alias} {TCP port number} {timeout value}
```

where:

SERVER ALIAS—Identifies the target server in the host file. Note that this name differs from the server name created in Step 1.

TCP PORT NUMBER—Identifies the target TCP port number on the target server.

TIMEOUT VALUE—Indicates period between connection attempts.

For example, for server test, TCP port 2001 and zero timeout, add:

```
/usr/lib/lp/model/standard "$@" | /tcp/crnl |
/tcp/tcp send test 2001 0
```

- or, from within the `/usr/lib/lp/node` directory, use both the `echo` function and a pointer to add this command line to the target filename.

Execute:

```
# echo /usr/lib/lp/model/standard "$@" | /tcp/crnl
| /tcp/tcpshd {server name} {TCP port number}
{timeout value} > {file name} [ENTER]
```

where:

FILENAME—Is the same name as that of the target queue. The pointer creates a file bearing that filename. For example, for server test, TCP port 2001 and zero timeout, pointing to queue name *queue1*, execute:

```
# echo /usr/lib/lp/model/standard "$@" | /tcp/crnl |
/tcp/tcpshd test 2001 0 > queue1 [ENTER]
```

10. Create both a print queue and a queue test. From the root directory, execute:

```
# /usr/lib/lpadmin -v/dev/null -ptest -mtest -u allow:all
-obanner [ENTER]
```

This command creates a banner preceding the printed material. To omit the banner, replace *-obanner* with *-onobanner*.

11. Start the Scheduler. Execute:

```
# /usr/lib/lpsched [ENTER]
```

12. Enable the queue test. Execute:

```
# enable test [ENTER]
```

13. Accept requests to test. Execute:

```
# accept test [ENTER]
```

14. Send a test file to the target printer. Execute:

```
# lp -d{queue name} {filename} [ENTER]
```

where:

QUEUE NAME—Identifies the target queue name.

FILENAME—Identifies a job to be printed; can be any file name.

For example, for target queue *test* and file name *file1*, execute:

```
# lp -dtest file1 [ENTER]
```

SCO_UNIX systems

1. Log into the UNIX host as root user.
2. From the root directory, create a directory called `tcp`. Execute:

```
# mkdir tcp [ENTER]
```

3. Copy files from the TCPSEND printing software program into `/tcp`. Execute:

```
# cp README /tcp [ENTER]
# cp Makefile /tcp [ENTER]
# cp tcpsend.c /tcp [ENTER]
# cp crnl.c /tcp [ENTER]
# cp conffll.model /tcp [ENTER]
```

4. Makefile is divided into sections pertaining to each of the supported types of UNIX systems (e.g., SCO, SUNOS, AIX, etc.). Within the file, locate the type of UNIX system you are using and un-comment the lines of code within that section. Those lines of code will become part of the libraries that are executed.

Locate the heading "For SCO System V 3.2" and un-comment these lines:

```
CFLAGS = -DSOCKET -DLAI_TCP -Di386
LIBS = -lsocket
```

5. Execute the **Make** command to compile the program. The path must point to the libraries. If your login doesn't have the correct path, the **Make** will not find the path of the libraries. Be sure that no errors occur during the compiling process. Execute:

```
# make [ENTER]
```

6. Before you configure a printer that affects the scheduler, stop the scheduler. Execute:

```
# /usr/lib/lpshut [ENTER]
```

7. Change both the ownership of the file and its group ID. This is done both for security reasons and to let all users execute the program. If ownership is not set up correctly, some users may not be authorized to print. Execute:

```
# chown lp tcpsend crnl [ENTER]
# chmod 555 tcpsend [ENTER]
# chgrp lp tcpsend [ENTER]
# chgrp lp crnl [ENTER]
```

8. Locate the UNIX system directory in which the printer interfaces reside. (Typically, this is the `/usr/spool/lp/model` directory.) Create a file in this directory using the same name as that of the queue. There are two ways to do this:

- Either use your favorite line editor to add the following command line to this file:

```
/usr/lib/lp/model/standard "$@" | /tcp/crnl | /tcp/tcpsend
{server name} {TCP port number} {timeout value}
```

where:

SERVER NAME—identifies the target server.

TCP PORT NUMBER—identifies the target TCP port number on the target server.

TIMEOUT VALUE—indicates period between connection attempts.

For example, for server *test*, TCP port *2001* and *zero* timeout, add:

```
/usr/lib/lp/model/standard "$@" | /tcp/crnl |
/tcp/tcpsend test 2001 0
```

- or, from within the `/etc/lp/model` directory, use both the *echo* function and a *pointer* to add this command line to the target filename. Execute:

```
# echo /usr/lib/lp/model/standard "$@" | /tcp/crnl
| /tcp/tcpsend {server name} {TCP port number}
{timeout value} > {file name} [ENTER]
```

where:

FILENAME—Is the same name as that of the target queue. The pointer creates a file bearing that filename. For example, for server *test*, TCP port *2001* and *zero* timeout, pointing to queue name *queue1*, execute:

```
# echo /usr/lib/lp/model/standard "$@" | /tcp/crnl |
/tcp/tcpsend test 2001 0 > queue1 [ENTER]
```

9. Check that the terminal server's IP address is in the `/etc/hosts` file. For example:

```
1.2.3.4 test VISTA
```

10. Create a print queue and a queue test. From the root directory, execute:

```
# /usr/lib/lpadmin -v/dev/null -ptest -mtest -u
allow:all -obanner [ENTER]
```

This command creates a banner preceding the printed material. To omit the banner replace *-obanner* with *-onobanner*.

11. Start the Scheduler. Execute:

```
# /usr/lib/lpsched [ENTER]
```

12. Enable the queue test. Execute:

```
# enable test [ENTER]
```

13. Accept requests to test. Execute:

```
# accept test [ENTER]
```

14. Send a test file to the target printer. Execute:

```
# lp -d{queue name} {filename} [ENTER]
```

where:

QUEUE NAME—Identifies the target queue name.

FILENAME—Identifies a job to be printed; can be any file name.

For example, for target queue *test* and file name *file1*, execute:

```
# lp -dtest file1 [ENTER]
```

AIX (BSD) systems

1. Log into the UNIX host as root user.
2. From the root directory, create a directory called *tcp*. Execute:

```
# mkdir tcp [ENTER]
```

3. Copy files from the TCPXFER printing software program into */tcp*. Execute:

```
# cp README /tcp [ENTER]
# cp Makefile /tcp [ENTER]
# cp tcpxfer.c /tcp [ENTER]
# cp crnl.c /tcp [ENTER]
# cp confll.model /tcp [ENTER]
```

4. Makefile is divided into sections pertaining to each of the supported types of UNIX systems (e.g., SCO, SUNOS, AIX, etc.). Within the file, locate the type of UNIX system you are using and un-comment the lines of code within that section. Those lines of code will become part of the libraries that are executed.

Locate the heading “For AIX” and un-comment these lines:

```
CFLAGS = -DSELECT -DSOCKET -DTABLESIZE -DAIX LIBS =
```

5. Determine the process ID of the qdaemon process. Execute:

```
# ps -ef [ENTER]
```

6. Kill the process ID of the qdaemon process on the system. Execute:

```
# kill -9 {process ID of qdaemon process} [ENTER]
```

7. Execute the **Make** command to compile the program. The path must point to the libraries. If your login does not have the correct path, the **Make** will not find the path of the libraries. Be sure that no errors occur during the compiling process. Execute:

```
# make [ENTER]
```

8. Change both the ownership of the file and its group ID to the appropriate spooling system ownership. This is done for security reasons and to let all users execute the program. If ownership is not set up correctly, some users may not be authorized to print.

9. Create a named pipe. A named pipe is a virtual device. Execute:

```
# cd /dev [ENTER]
# mknod {named pipe} p [ENTER]
```

where:

dev—In most UNIX systems, this is a directory where all devices are kept.

NAMED PIPE—Identifies the virtual device. For example the named pipe might be *queue1*.

10. Create a print queue. Using your favorite text editor, add the printer you want to create. You can copy the section of an existing queue and rename it with the name of the target printer. For example, to use the UNIX *vi* editor, execute:

```
# vi /etc/qconfig [ENTER]
```

where:

etc/qconfig—is the file IBM uses to store all of the printer definitions. It is similar to an */etc/printcap* file.

11. Create a virtual printer device. Execute:

```
# mkvirprt -d{named pipe} -n{device} -q {queue name}
-s asc -t printer [ENTER]
```

where:

- d—Is the full path of named pipe being used.
- n—Device used (in `/etc/qconfig`)
- q—Select any name and assign it to the queue
- s—Indicates the type of data stream (use `asc` for ASCII)
- t—Indicates the printer type (use `printer` for standard printers)

12. Remove the old printer binary file. All of the system-related files that IBM uses are binary files. For printing, IBM uses `qconfig.bin`. This is a system-created file that cannot be edited. Since you can't edit the file, you must remove it. Execute:

```
# rm /etc/qconfig.bin [ENTER]
```

13. Rebuild the `qconfig.bin` file containing the revisions you made. Execute:

```
# enq -d [ENTER]
```

14. Start all the queues in the `/etc/rc.local` file. In most systems, at startup, there are `rc` (run command) files that start the daemons for multi-users, E-mail, Telnet and other operations, including the TCPXFER software. To automatically start the TCPXFER daemons when the system reboots, execute:

```
# if [ -f /usr/local/bin/tcpxfer ]; then [ENTER]
# /usr/bin/nice /usr/local/bin/tcpxfer /dev/ptyre
  {alias name} {TCP port number} & [ENTER]
# /usr/bin/nice /usr/local/bin/tcpxfer /dev/ptyrf
  {alias name} {TCP port number} & [ENTER]
```

where:

usr/bin/nice—The **TCPXFER** command is a daemon running in the background of the system. Starting this daemon with `/usr/bin/nice` gives the TCPXFER program a lower priority, which allocates less CPU time to TCPXFER but allocates more CPU time to process other user commands.

ptyre and **ptyrf**—These are the named pipes. They are variable names.

ALIAS NAME—This is the alias name specified in the `/etc/hosts` file, which lists all of the hosts and devices in the network. The alias identifies the target server file name that is associated with the target printer.

For example, if the alias name is *vista* and the corresponding IP address (which must be in the `/etc/hosts` file) is *1.2.3.4*, the associated network

service name (also from the `/etc/hosts` file) might be *Vista*. You need not use the alias; you can substitute the IP address of the server, if desired. In this example, you could substitute *1.2.3.4* for *vista*.

TCP PORT NUMBER—Specifies the virtual port number (or socket) of the target TCP port; for example, *2001*.

&—The ampersand instructs the system to start TCPXFER in the background if the system gets rebooted.

If you don't want to reboot the entire system at this time, you can also manually restart the queue daemons. However, when you reboot the host processor, these programs won't exist, so you will have to re-execute these commands. For every named pipe associated with a queue, execute:

```
# /usr/bin/nice /usr/local/bin/tcpxfer /dev/{named pipe}
{alias name} {TCP port number} & [ENTER]
```

15. Start the scheduler. Execute:

```
# qdaemon & [ENTER]
```

16. Submit a test print to confirm that the system is set up properly.

Configuring for LAT Protocol

Following are procedures to connect to remote LAT services and create local LAT services. Each procedure is described briefly. For further information about command parameters used in each command, see appendix A.

Connecting to Remote Services

Activating Automatic Logout Features—Network Services

Following is a procedure to configure a server port to allow a user to log into that port and connect to all services. When the terminal is either idle or powered *off*, **the user's port is automatically logged out.**

1. Physically attach an interactive terminal to a serial port on the server.
2. Log into the server.

Note: If the target port (that is, the server port that you are configuring) is dedicated to a service, you cannot log into that port without automatically connecting to that service. Therefore, to **Change**, **Define** or **Set** characteristics on a dedicated port, you must log into another port (one that is *not* dedicated to a service).

3. Set the port through which you are operating to *Privileged* status.

```
Local>      set privileges [ENTER]
Password>   {privileged password} [ENTER]
```

4. Execute the following port configuration commands:

```
Local>      change port {number of target port} access dedicated
             none dsrlogout enabled inactivity logout enabled
             [ENTER]
```

Refer to appendix A for more information on the command variables.

5. Log out of the target port.

```
Local>      logout port {number of target port} [ENTER]
```

Automatically Connecting a Server Port to a Service

Attach an interactive terminal to a server port. When you log into this port, the server automatically bypasses the local mode and connects you to a single Ethernet service.

Note: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that already is dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

1. At the Local> prompt, execute:

```
Local>      change port {number of target port} access autoconnect
             enabled break disabled dedicated {name of service}
             [ENTER]
```

Note: If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** and **Define** commands. However, if you are operating through the target port, which is being configured, you can execute only the **Define** command.

2. To authorize the appropriate groups, execute:

```
Local>      change port {number of target port} authorized
             groups {list of group numbers} [ENTER]
```

Note: If the same network service is offered by two different group numbers, assigning authorized groups to a port ensures that the specified port on the server can communicate with either of the two groups that offer the service.

3. Log out of the target port.

```
Local>  logout port {number of target port} [ENTER]
```

Disabling the Autoconnect Feature

If a server port that is currently dedicated to a service, must be changed to a port that has access to multiple services, log into another port on the server, and at the Local> prompt execute the following commands, in sequence:

```
Local>  change port {number of target port} autoconnect
        disabled [ENTER]
```

```
Local>  change port {number of target port} dedicated none
        [ENTER]
```

```
Local>  logout port {number of target port} [ENTER]
```

Creating Local LAT Services

Matching Each Server Port to its Device

You can create local LAT services, such as modems and printers, and offer them through Line Card ports on the server. To establish a local service, select a server port and match the data flow parameters of that port to the requirements of the attached device that is being offered as a local service. Follow this procedure to assure the most basic port and service characteristics are determined.

1. Physically attach the device (that will be offered as a service) to a port on the server.
2. Log into the server.
3. Execute the following commands:

```
Local>  change service {name of new LAT service} port {server
        port offering LAT service} lat enabled identification
        "{short description of service}" [ENTER]
```

Use the LAT service name when you execute the **Connect** {name of service} command.

Note: You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that exactly the same service is offered on each node.

4. Log out of the target port:

```
Local>  logout port {number of target port} [ENTER]
```

Defining a LAT Service Password

To limit user access to a local service, create a service password (up to six ASCII characters) and assign it to the service as follows:

1. Return to the `Local>` prompt and execute the command:

```
Local>      change service {name of LAT service} password  
[ENTER]
```

2. When the server displays the `Password>` prompt, type:

```
Password>   {service password} [ENTER]
```

3. The server displays the `Verification>` prompt before adding the new service password to its database. Type:

```
Verification> {service password} [ENTER]
```

Testing New LAT Services

Upon adding a new LAT service to the network, check the correctness of the installation by attempting to successfully connect to the service. If you are able to connect to the service, continue your test by sending operating commands to the device. If the device responds properly, execute either the **Break** or **Switch** command to return to the local mode. Then, execute the **Disconnect Session** command to terminate the connection to the service.

Note: If the new service is offered on a non-LAT host processor, repeat the test procedure to confirm that the first session has been disconnected.

1. If you cannot connect to the service, or, if you can connect but the device does not respond properly, execute:

```
Local>      show service {name of LAT service} characteristics  
[ENTER]
```

2. Compare the displayed listing of ports assigned to the service with your listing of the server ports you believe are assigned to the service. Determine if the proper ports are assigned to the service. Correct the port assignments, if necessary.
3. Examine the port characteristics. Compare the displayed listing of target port characteristics with your listing of the characteristics you believe are assigned to that port. Determine if the target port has been correctly configured to match the characteristics of the attached device. Correct the target port characteristics, if necessary. Execute:

```
Local>      show port {number of target port} characteristics  
[ENTER]
```

Configuring the Server for Printer Support**XON/XOFF Flow Control**

Following is an example of the configuration required to support a printer (such as an HP LaserJet) as a local LAT service. When the target server port is properly configured, the LaserJet printer will be accessible to all network users.

1. Physically attach the HP LaserJet printer to a Line Card port.
2. Log into the server.
3. Refer to the following notes before executing the port commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** or **Define** commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that the identical service is offered on each node.

```
Local> change port {number of target port} access remote
autobaud disabled break disabled dedicated none
dsrlogout enabled inactivity logout enabled name
{name of target port} [ENTER]
```

NUMBER OF TARGET PORT—Server port that offers the LAT service. For information on other command parameters, see appendix A.

4. Examine the port characteristics. Compare the displayed listing of target port characteristics with your listing of the characteristics you believe are assigned to that port. Determine if the target port has been correctly configured to match the characteristics of the attached device. Correct the target port characteristics, if necessary. Execute:

```
Local show port {number of target port} characteristics
[ENTER]
```

5. Log out of the target server port:

```
Local>  logout port {number of target port} [ENTER]
```

Note: It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

6. Assign a LAT service name to the printer port. Create an identifying message for the service. Execute:

```
Local>  change service {name of service} port {number of
printer port} identification "{identifying string}"
[ENTER]
```

7. To confirm the newly assigned port parameters, execute:

```
Local>  show port {number of target port} characteristics [ENTER]
```

8. Compare the displayed listing of target port characteristics with your listing of the characteristics you believe are assigned to that port. Determine if the target port has been correctly configured to match the characteristics of the attached device. Correct the target port characteristics if necessary.

CTS/RTS Flow Control

Attach a printer to the server. The printer receives data only when it sends a Clear-to-Send (CTS) modem signal to either the server port or the attached interactive terminal. The printer sends its CTS signal in response to a Request-to-Send (RTS) flow control signal that is generated by either the server port or the attached terminal. For cabling information, see the *Communications Server Installation and Service Guide*. Refer to the following notes before executing the port configuration commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** and **Define** commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that the identical service is offered on each node.

It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

1. Configure the characteristics of a target port on the server for the CTS/RTS printer. Execute:

```
Local>  change port {number of target port} access remote
        autobaud disabled autoprompt disabled dsrlogout
        enabled flow control CTS handshake CTS inactivity
        logout enabled modem control disabled speed {printer
        baud rate} name {name of port} [ENTER]
```

For information on command parameters, see appendix A.

2. After you successfully configure the target port for the CTS/RTS printer, log out of the target port. Execute:

```
Local>  logout port {number of target port} [ENTER]
```

3. Next, assign a service name to the target port. Create an identifying message for the service. Execute:

```
Local>  change service {name of LAT service} port {number of
        target port} identification "{identifying string}"
        [ENTER]
```

DSR/DTR Flow Control

Attach a printer or a printing terminal to the server. The cable that connects the terminal device to the server port changes the incoming Data-Terminal-Ready (DTR) signal to an outgoing Data-Set-Ready (DSR) signal. The change works in both directions. That is, when the server port is ready to receive data from the interactive terminal, it sends a DTR signal through the cable. The DTR signal becomes a DSR signal. When the terminal at the receiving end of the cable detects DSR, it begins transmitting data to the server port at the other end. When the terminal completes its transmission, it sends a DTR signal through the cable. This becomes a DSR signal at the server, and the server begins transmitting to the terminal. For cabling information, see the *Communications Server Installation and Service Guide*. Refer to the following notes before executing the port configuration commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** and **Define** commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that the identical service is offered on each node.

It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

1. Configure the characteristics of a target port on the server for the DSR/DTR printer or printing terminal. Execute:

```
Local>  change port {number of target port} access remote
        dsrlogout disabled flow control dsr inactivity logout
        enabled modem control disabled speed {printer baud
        rate} name {name of target port} [ENTER]
```

For information on command parameters, see appendix A.

2. After you have successfully configured the target port characteristics, log out of the target port. Execute:

```
Local>  logout port {number of target port} [ENTER]
```

3. Next, assign a service name to the target port. Create an identifying message for the service. Execute:

```
Local>  change service {name of LAT service} port {number of
        target port} identification "{identifying string}"
        [ENTER]
```

Configuring the Server for Modem Support

Setting up a Dial-out Modem Pool

A dial-out modem attached to a server port allows server users to log into the port and be connected to a modem. Users then can dial out through that modem and connect to a second modem which can be attached to a host or an interactive device on a distant network.

Following is a procedure to attach a dial-out modem to a server port and to offer the dial-out modem as a LAT service. Refer to the following notes before executing the port configuration commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** and **Define** commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that the identical service is offered on each node.

It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

1. Attach the modem to a port in the server Line Card.
2. Log into the server.

3. Execute the following port configuration commands:

```
Local>  change port {number of target port} access remote
        autobaud disabled autoprompt disabled break disabled
        dsrlogout disabled dtrwait enabled inactivity logout
        disabled dedicated none modem control enabled [ENTER]
```

NUMBER OF TARGET PORT—Server port offering the LAT service. For information on other command parameters, see appendix A.

4. After you have successfully configured the target port characteristics, log out the target port. Execute:

```
Local>  logout port {number of target port} [ENTER]
```

5. Next, assign a service name to the target server port. Create an identifying message for the service. Execute:

```
Local>  change service {name of LAT service} port {number of
        port} identification "{identifying string}" [ENTER]
```

6. To create a security password for the modem service, execute:

```
Local>  change service {name of LAT service} password
        {password string} [ENTER]
```

Setting up for Dial-in/Dial-out Modems

Attach a dial-in/dial-out modem to a port configured for dynamic access on the server. Dial-in/dial-out modems can both initiate and receive telephone calls. Offer the modem as a service on the Ethernet. The dual-function modem allows access to the server by both local and remote interactive users. Users must log out the port to switch between dial-in and dial-out modes of operation. Refer to the following notes before executing the port configuration commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes

multicast the same LAT service name, the network server assumes that the identical service is offered on each node.

It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

1. Configure the characteristics of a target port on the server for the dial-in/dial-out modem. Execute:

```
Local>  change port {number of target port} access
        dynamic autobaud disabled dsrlogout disabled dtrwait
        enabled flow control xon inactivity logout enabled
        modem control enabled password enabled input speed
        {rate of speed} output speed {rate of speed} [ENTER]
```

For information on command parameters, see appendix A.

2. After you have successfully configured the target port characteristics, log out the port. Execute:

```
Local>  logout port {number of target port} [ENTER]
```

3. Next, assign a service name to the target port. Create an identifying message for the service. Execute:

```
Local>  change service {name of LAT service} port {number of
        target port} identification "{identifying string}"
        [ENTER]
```

4. To create a security password for the modem service, execute:

```
Local>  change service {name of LAT service} password
        {password string} [ENTER]
```

Using the Server as a Front End to a Non-LAN Host

Some devices are not equipped for communication via the Ethernet. Devices that cannot be directly connected to the LAN instead must operate through an interface device (such as a server) which, in turn, is connected to the network. In such cases, the server operates as a front end to the non-LAN host processor. By defining the non-LAN host as a local service, the host becomes available to network users. Following is a procedure to configure a non-LAN host processor as a service on the server. Refer to the following notes before executing the port configuration commands that are described in the numbered section following the notes.

Notes: You cannot log into a port that is dedicated to a service without connecting to that service. Therefore, to **Change**, **Set** or **Define** the Port Dedicated characteristic for a target port that is already dedicated to a service (but that is not presently connected to that service), you must log into another port (one that is not dedicated to a service) and execute the **Dedicated None** command for the desired port from the other port.

If you are operating through a port other than the target port that is being configured for autoconnection, you can execute the **Change**, **Set** or **Define** commands. However, if you are operating through the target port that is being configured, you can only execute the **Define** command.

You must confirm the uniqueness of a service name before assigning it to a service offered on a server port. Recall that if several nodes multicast the same LAT service name, the network server assumes that exactly the same service is offered on each node.

It is possible to create a LAT service name and assign that name to all of the ports that are configured for that local service. That way, users can simply connect to the service (without specifying a port) and obtain a connection to the service through the next available port. Execute step six to create a LAT service name pointing to the port.

1. Physically attach the non-LAN host to a server Line Card port.
2. Log into the server.
3. Execute the following port configuration commands:

```
Local>  define port {number of target port} access
        remote autobaud disabled autoprompt disabled break
        disabled dtrwait enabled inactivity logout disabled
        dedicated none modem control enabled [ENTER]
```

NUMBER OF TARGET PORT—Server port that offers the non-LAN host service. For information on other command parameters, see appendix A.

4. Log out of the target port configured to connect to the non-LAN host.

```
Local> logout port {number of target port} [ENTER]
```

5. Confirm that the target port settings are properly recorded. Execute:

```
Local> show port {number of target port} characteristics [ENTER]
```

6. Assign a service name to the server port. Also, create an identifying message for the service. Users can execute the **Connect** {service name} command to start a session with the non-LAN host. Execute:

```
Local> change service {name of service} port {number of target port} identification "{identifying string}" [ENTER]
```

Configuring a Static PPP Link

This section explains how to configure server ports to use Point-to-Point Protocol (PPP). The objective is to provide a more direct route to selected IP hosts via a static PPP link; that is, a PPP link that does not dynamically reestablish itself if the link is terminated.

In the static PPP link configuration (figure 2-12), Server1 will connect Host1 to Host2 through Server2. The hosts are located on separate Ethernets. Server1, on Host1's Ethernet, is connected via a static PPP link to Server2, which is on Host2's Ethernet. The serial ports connecting the Servers are gateways through their respective servers. Each gateway port is assigned a unique "gateway" IP address.

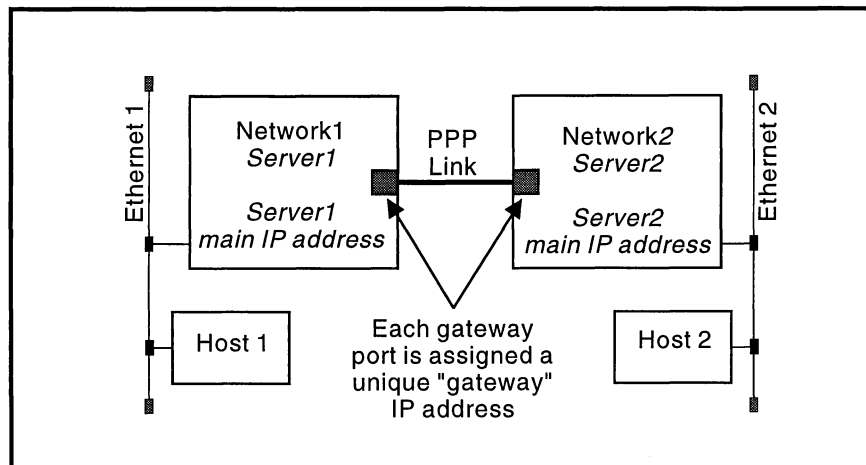


Figure 2-12. Configuring servers for PPP

When you establish a link between local and remote servers, coordinate all aspects of the PPP link configuration with personnel at the remote location. To configure the servers for PPP, proceed as follows:

1. Assign privileges to your port. Execute:

```
Server1>    set privileged [ENTER]
```

2. Supply the privileged password. Execute:

```
Password>   {password string} [ENTER]
```

3. Assign Server1 a main IP address. Execute:

```
Server1>    change address {main IP address of Server1}  
            mask {subnet mask, if applicable} [ENTER]
```

4. Determine available interface numbers at Server1. Execute:

```
Server1>    show interfaces [ENTER]
```

5. Select an available interface number. Assign a gateway IP address to the interface dedicated to the PPP link. Execute:

```
Server1>    change address {IP address of PPP interface}  
            interface {interface number} mask {subnet  
            mask, if applicable} [ENTER]
```

6. Assign characteristics to the interface port. Execute:

```
Server1>    change port {port number} access dynamic  
            autobaud disabled flow control disabled [ENTER]
```

7. Assign the PPP link to a Server1 port. Execute:

```
Server1>    change ppp link {port number dedicated to link}  
            state open [ENTER]
```

8. Assign the PPP link to the interface. Execute:

```
Server1>    change interface {interface number} port {port  
            number} type ppp state up autoinit enabled  
            compression disabled [ENTER]
```

9. In preparation for creating a route to Server2, determine available route numbers. Execute:

```
Server1>    show routes [ENTER]
```

10. Create a route to the destination server, and change to that route. Execute:

```
Server1>  change route {route number} destination  
          {Server2's ppp link gateway address} gateway  
          {Server1's gateway IP address} interface  
          {interface number} mask {subnet mask if  
          applicable} type network direct enable [ENTER]
```

11. Reboot the server. Execute:

```
Server1>  crash [ENTER]
```

12. Determine the location of Server2. Coordinate the configuration procedure with personnel at the remote location.

13. At Server2, assign privileges to your port. Execute:

```
Server2>  set privileged [ENTER]
```

14. Supply the privileged password. Execute:

```
password> {Privileged password string} [ENTER]
```

15. Assign Server2 a main IP address. Execute:

```
Server2>  change address {main IP address of Server2} mask  
          {subnet mask, if applicable} [ENTER]
```

16. Determine available interface numbers at Server2. Execute:

```
Server2>  show interfaces [ENTER]
```

17. Select an available interface number. Assign a gateway IP address to the interface dedicated to the PPP link. Execute:

```
Server2>  change address {IP address of PPP interface}  
          interface {interface number} mask {subnet  
          mask, if applicable} [ENTER]
```

18. Assign characteristics to the interface port. Execute:

```
Server2>  change port {port number} access dynamic  
          autobaud disabled flow control disabled [ENTER]
```

19. Assign the PPP link to a Server2 port. Execute:

```
Server2>  change ppp link {port number dedicated to  
          link} state open [ENTER]
```

20. Assign the PPP link to the interface. Execute:

```
Server2>    change interface {interface number} port  
            {port number} type ppp state up autoinit  
            enabled compression disabled [ENTER]
```

21. Determine available route numbers. Execute:

```
Server2>    show routes [ENTER]
```

22. Create a route to Server1, and change to that route. Execute:

```
Server1>    change route {route number} destination {Server1's  
                ppp link gateway address} gateway {Server2's  
                gateway IP address} interface {interface number}  
                mask {subnet mask if applicable} type network  
                direct enable [ENTER]
```

23. Enable Server2 to forward IP packets to the Ethernet. Execute:

```
Server2>    change ip forward enabled [ENTER]
```

24. Associate the IP address of the PPP interface port for Server1 with the Ethernet address of Server2. Execute:

```
Server2>    change arp {ip address of PPP interface at Server1}  
                ethernet {ethernet address of server2} proxy  
                enabled [ENTER]
```

25. Reboot Server2. Execute:

```
Server2>    crash [ENTER]
```

26. Start the interface connection. Execute:

```
Server2>    start interface {interface number at server2}  
                [ENTER]
```

You now have a static PPP link between Server1 and Server2.

Note: To support PPP links with remote hosts other than another server, such as a PC, the remote host must have an IP address. See figure 2-13.

For further information about temporarily assigning an IP address to a remote host, see *Managing the IP Address Pool*, in chapter 4.

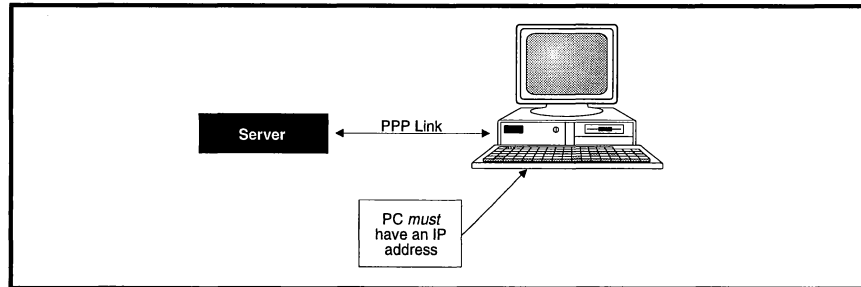


Figure 2-13. Connecting server to PC host via PPP link

Configuring a Dynamic PPP Link

The server supports both static and dynamic PPP interfaces. Recall that static interfaces remain dedicated (via hard-wire) to the PPP interface at all times. server ports that are dedicated to the static PPP interface cannot be used for other interfaces, such as Ethernet connections.

Dynamic PPP interfaces only become dedicated to PPP lines on demand. Thus, when a PPP line is not active at the server port, that port is available for connections to other interfaces. When a PPP line is needed at the server port, the user executes a short configuration procedure to start the PPP interface.

A useful application for dynamic PPP is a PC user running software that supports TCP/IP. Such software permits the user to transfer files, exchange mail and act as a Telnet host.

By reconfiguring the line connecting the PC and the server from asynchronous operation to a PPP link, the server becomes an IP gateway and the PC becomes an IP host. Note that the PPP line need not be a solid wire; instead, each device can be connected to a modem. See figure 2-14.

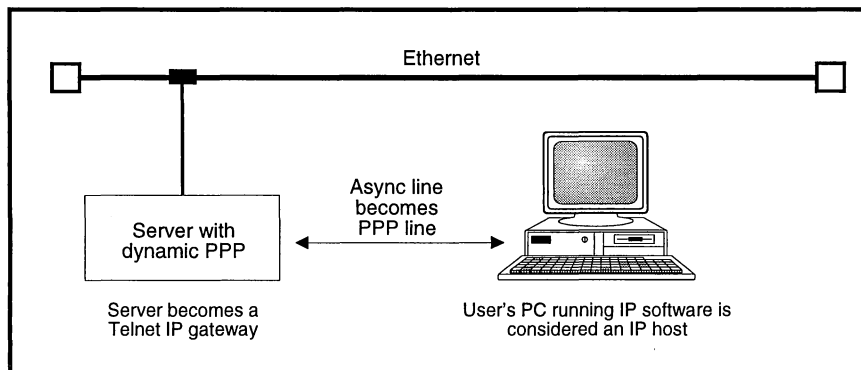


Figure 2-14. Changing async line to PPP line

To configure a dynamic PPP line connecting a user's PC with the server, proceed as follows:

1. The PC must contain a terminal emulation program, such as *HBScm*, which permits the PC to access the server. When that application is loaded into the PC, start the program and connect to the server port.
2. Upon connecting, awaken the target server port and obtain the > prompt. Execute:

[ENTER] [ENTER]

The local server prompt appears.

Local>

3. Confirm that the following port parameters are activated:

- Port Access Dynamic
- Port Flow Control Disabled
- Port Autobaud Disabled

Execute:

Local> **show port characteristics [ENTER]**

4. Change the port parameters to the required settings, if necessary. Execute:

Local> **set port access dynamic flow control disabled autobaud disabled [ENTER]**

5. Decide whether you will use the server's main IP address during PPP negotiations, or an IP address from the server's pool of IP addresses. Consider the following instances:
 - If you are the only user (or if there are relatively few users), starting a dynamic PPP connection, use the Communications Server's main IP address for PPP negotiations.
 - If many users will be starting dynamic PPP links, it may not be practical to assign an IP address to every user making a connection. Instead, you can create a pool of IP addresses which the server can use during PPP negotiations. For more information, see *Managing the IP Address Pool* in chapter 4.

Note: If you are using *PCTCP*, *KA9Q* or a similar product, you may have an IP address that is "hard coded" into the server; that is, an IP address that is permanently assigned to the server. If so, you will not be able to initiate a dynamic PPP link using the IP pool of addresses. For more information, refer to the user documentation specific to your application.

- 6a. Start the PPP link using the server's main IP address. Execute:

```
Local> connect ppp {main IP address} [ENTER]
```

- 6b. Start the PPP link using the next available address in the pool of IP addresses. Execute:

```
Local> connect ppp 0.0.0.0 [ENTER]
```

7. When the connection is established, a message appears at the user's PC:

```
Session Established
```

8. Exit from the terminal emulation program and return to the PC DOS prompt.

Note: The syntax of the **exit** command or the assigned **EXIT** function key depends upon the terminal emulation package installed in the PC. Check your application user documentation for further information.

For example, if you are using the HBScom terminal emulation program, execute:

```
HBSCOM> exit [ENTER]
```

9. The PC screen should display the DOS prompt.

```
C:\>
```

10. Start the the PC-resident Telnet application software program which configures the PC as a Telnet host. Execute:

```
C:\> {start command} [ENTER]
```

START COMMAND—The character string which activates the PC-resident Telnet application software. For example, if **net** is the command used to start the application, execute:

```
C:\> net [ENTER]
```

11. At the application prompt, connect to the desired destination host using either the IP address of the host or the name of the host (by query to the domain nameserver):

```
PROMPT> connect Telnet {IP address of destination host} [ENTER]
```

or

```
PROMPT> connect Telnet {name of host} [ENTER]
```

12. To end the Telnet session, break out of the host and return to the application prompt.

```
HOST PROMPT> [BREAK]
```

13. You are returned to the local prompt of the server. Either continue or terminate the session. Execute:

```
Local> disconnect {number of session} [ENTER]
```

Compressed PPP

The strategy of compressed PPP is to improve data throughput on PPP lines. Throughput improvement is achieved by shortening the TCP/IP header, which is part of every packet. Recall that during the typical interactive Telnet session only one byte of data is sent across the internet. However, the TCP and IP headers required to send that single byte, together consume at least 40 bytes. When a serial link is used to exchange the data packets and acknowledgments, two packets are typically required for that single byte of data. First, there is the data packet containing 41 bytes, and then there is a data acknowledgment packet containing 40 bytes, which returns from the destination host to the source.

The resulting traffic ties up the point-to-point link, reducing the typing speed bandwidth that should be available given the speed of the link. Unlike a hardwired, 9600 baud connection, in which the total 9600 bytes are available for data transfer, here only $\frac{1}{40}$ th of the total capacity of a hardwired 9600 baud connection is available.

Experts estimate the value, $\frac{1}{40}$ th, because the TCP/IP connection is a full duplex connection. In such a connection, the returning acknowledgment packet can pass the next outgoing new packet. Obviously, $\frac{1}{40}$ th of 9600 baud isn't very fast, especially when you realize that the objective is to transfer a single byte of data across the link and that 40 bytes of header are consumed in that process.

In PPP the problem is magnified because additional characters control the frame *startup*, *stop* and *escape* functions. Note that this same problem occurs in other protocols, such as Serial Line Internet Protocol (SLIP). Therefore, the header compression solution used in PPP links applies equally to other link level protocols, such as SLIP. TCP/IP packet headers are compressed using a technique called the Van Jacobson header compression algorithm. The same algorithm used to compress the packet headers is also used to expand them upon arrival at the destination host.

The server implements this compression algorithm as *Compressed PPP*, which uses the Van Jacobson header compression algorithm with the PPP extensions as described in RFC1144. Compression in a PPP interface can be Active, Enabled or Disabled:

- If you set compression to *Active*, all TCP packets compressible according to the RFC1144 specification are sent compressed. In this mode, the server can receive all normal TCP/IP packets and compressed or non-compressed

headers. Typically, 80-95% of headers are compressed down from 40 bytes to between 3 and 5 bytes.

- If you set compression to Enabled, the server doesn't send any compressed packet headers, but it can recognize them. Upon detecting incoming compressed headers, outgoing packet headers are compressed.
- If you set compression to Disabled, the server will not attempt to send compressed packet headers regardless of the contents of the packets it receives. This setting permits users to eliminate compression from the data transmission process, which is useful for debugging a link. To set a serial interface for Compressed PPP, execute:

```
Local> change interface {number of interface}
      type ppp port {number of port} state up
      compression {compression parameter} [ENTER]
```

PPP Authentication Protocols

Point-to-Point Protocol (PPP) is a standard for transferring data across both synchronous and asynchronous serial links. It was devised to fill the need for a robust serial link protocol—one with many configurable features—that could manage datagrams in multiple protocols and deliver them across a single link.

Due to its largely successful implementation, PPP can be found on many serial data links between users and hosts. For example, many commercial users trade information across town and across continents via serial links using PPP. Corporate host processors containing files of sensitive financial, marketing and product development information now can be shared across the network by many users. As more network resources become available to a larger user base, tighter security is required to prevent unauthorized access to sensitive files. In response to this need, the stand-alone servers support two authentication protocols which enhance the security of PPP links:

- **Password Authentication Protocol (PAP)**—in which the stand-alone server must successfully exchange username and password information with the target host before the server user can access that host. Once access is allowed, no further identification checks are required.
- **Challenge Handshake Authentication Protocol (CHAP)**—which provides a greater measure of link security than PAP. With CHAP, the target host periodically challenges the user's server to defend its right to access. At regular intervals after the link is established, the host sends encrypted messages to the server. The server must decrypt each message and respond with an appropriate acknowledgment. The encryption codes change with each challenge.

Sample PAP/CHAP Configuration

To use either of these authentication protocols on a PPP link, first, set up the PPP link. Then, assign the authentication parameters to both the local and remote ends of the link. In figure 2-15, two servers are communicating via a PPP link. Initially, both servers are configured to use any authentication protocol. Authentication protocol is easily changed to PAP, CHAP or TACACS on either or both of the servers.

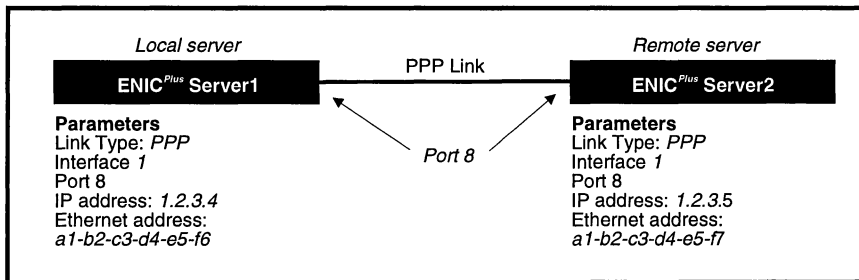


Figure 2-15. Two servers communicating via a PPP link

Part One: Configuring the Remote Server

1. Select a remote server port and assign a PPP link interface to that port. In this example, assign server port eight to server interface 1. Execute:

```
Server2> change interface 1 type ppp port 8 [ENTER]
```

2. Configure the remote server port parameters. Execute:

```
Server2> change port 8 access remote autobaud disable  
flow control disabled speed 9600 [ENTER]
```

3. Configure a route from the remote server to the local server. Execute:

```
Server2> change route 6 destination 1.2.3.4 gateway  
1.2.3.4 interface 1 type host [ENTER]
```

4. Configure the remote server to use proxy ARP. Execute:

```
Server2> change arp 1.2.3.4 ethernet a1-b2-c3-d4-e5-  
f6 proxy enabled mask 255.255.255.255 [ENTER]
```

5. Configure the PPP link data transfer parameters at the remote server. Note that all data transfer parameters are set to their default values. Magic Number, Protocol Compression and Quality Monitor all are disabled; the Frame Checking Sequence is 16; and the State of the link is closed. To reset all parameters to their default values, execute:

```
Server2> change ppp link 8 magic disabled quality disabled
proto_compress disabled addr_compress disabled fcs 16 state closed [ENTER]
```

6. Configure the PPP link authentication parameters at the remote server. In this example, the remote server is set to Any, which allows it to adapt to whatever authentication protocol is used at the local server. By default, if Any fails, the remote server will default to CHAP. If CHAP fails, the remote server will default to PAP. Authentication Protocol Table instructs the remote server to refer to its local database of usernames and passwords to verify users. The MRU values are set to their default settings. Execute:

```
Server2> change ppp link 8 authentication any
auth_protocol table mrumin 200 mrumax 1500 [ENTER]
```

7. Configure the PPP link to send ASCII values 0 to 31 to the local server during data transmission. Execute:

```
Server2> change ppp link 8 async_map 0-31 escape [ENTER]
```

Part Two: Configuring the Local Server

1. Select a local server port and assign a PPP link interface to that port. In this example, assign server port eight

to server interface 1. Execute:

```
Server1> change interface 1 type ppp port 8 [ENTER]
```

2. Configure the local server port parameters. Execute:

```
Server1> change port 8 access remote autobaud disable
flow control disabled speed 9600 [ENTER]
```

3. Configure a route from the local server to the remote server. Execute:

```
Server1> change route 6 destination 1.2.3.5 gateway
1.2.3.5 interface 1 type host [ENTER]
```

4. Configure the local server to use proxy ARP. Execute:

```
Server1> change arp 1.2.3.5 ethernet a1-b2-c3-d4-e5-f7
proxy enabled mask 255.255.255.255 [ENTER]
```

5. Configure the PPP link data transfer parameters at the local server. Note that all data transfer parameters are set to their default values. Magic Number, Protocol Compression and Quality Monitor all are disabled; the Frame Checking Sequence is 16; and the State of the link is closed. To reset all parameters to their default values, execute:

```
Server1>    change ppp link 8 magic disabled quality
             disabled proto_compress disabled addr_compress
             disabled fcs 16 state closed [ENTER]
```

6. Configure the PPP link authentication parameters at the local server. In this example, the local server is set to Any, which allows it to adapt to whatever authentication protocol is used at the remote server. By default, if Any fails, the remote server will default to CHAP. If CHAP fails, the remote server will default to PAP. Authentication Protocol Table instructs the local server to refer to its local database of usernames and passwords to verify users. The MRU values are set to their default settings. Execute:

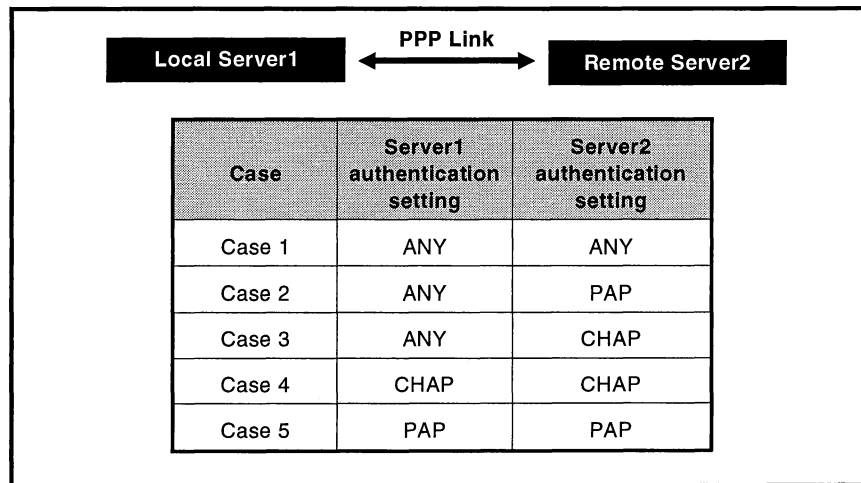
```
Server1>    change ppp link 8 authentication any auth_proto
             table mrumin 200 mrumax 1500 [ENTER]
```

7. Configure the PPP link to send ASCII values 0 to 31 to the remote server during data transmission. Execute:

```
Server1>    change ppp link 8 async_map 0-31 escape [ENTER]
```

Starting the Link

When both the local and remote servers are configured, you can start the link using several combinations of authentication protocols. Figure 2-16 illustrates five possible combinations. Each possibility is identified by a case number. To start the link using any of these configurations, refer to the procedures below. Locate the case number that matches your server-to-server configuration, and follow the indicated procedures.



Configuration

Figure 2-16. Possible PPP link authentication settings

Case 1: Both local and remote servers are set to *Any* authentication protocol.

1. Execute *Part Two: Configuring the Local Server*.
2. To start the link at the local server, execute:

```
Server1> start interface 1 [ENTER]
```
3. Execute *Part One: Configuring the Remote Server*.
4. To start the link at the remote server, execute:

```
Server2> start interface 1 [ENTER]
```

Case 2: Local server is set to Any authentication protocol; remote server is set to PAP.

1. Execute *Part Two: Configuring the Local Server*.

2. To start the link at the local server, execute:

```
Server1> start interface 1 [ENTER]
```

3. Execute *Part One: Configuring the Remote Server*.

4. At the remote server, reset the PPP link interface to use PAP. Execute:

```
Server2> change ppp link 8 authentication pap [ENTER]
```

5. At the remote server, activate the authentication feature at the port assigned to the link. Execute:

```
Server2> change port 8 access remote authentication  
enabled [ENTER]
```

6. At the remote server, create a new authentication file for the user (*Susan*) and assign a password (*28may60*) for that user. Set the authentication file protocol to PAPCHAP and indicate the IP address of the local server to which the user will connect. Associate the authentication file number with the server port number through which the link will operate. Execute:

```
Server2> change authentication 1 protocol papchap  
sername "susan" password "28may60" ip 1.2.3.4  
port 8 [ENTER]
```

7. At the remote server, confirm the authentication settings. Execute:

```
Server2> show authentication [ENTER]
```

8. Start the interface at the remote server. Execute:

```
Server2> start interface 1 [ENTER]
```

Case 3: Local server is set to Any authentication protocol; remote server is set to *CHAP*.

1. Execute *Part Two: Configuring the Local Server*.

2. To start the link at the local server, execute:

```
Server1> start interface 1 [ENTER]
```

3. Execute *Part One: Configuring the Remote Server*.

- At the remote server, reset the PPP link interface to use *CHAP*. Execute:

```
Server2> change ppp link 8 authentication chap [ENTER]
```

- At the remote server, activate the authentication feature at the port assigned to the link. Execute:

```
Server2> change port 8 access remote authentication
enabled [ENTER]
```

- At the remote server, create a new authentication file for the user (*Susan*) and assign a password (*28may60*) for that user. Set the authentication file protocol to *PAPCHAP* and indicate the IP address of the local server to which the user will connect. Associate the authentication file number with the server port number through which the link will operate. Execute:

```
Server2> change authentication 1 protocol papchap
username "susan" password "28may60" ip 1.2.3.4
port 8 [ENTER]
```

- At the remote server, confirm the authentication settings. Execute:

```
Server2> show authentication [ENTER]
```

- Start the interface at the remote server. Execute:

```
Server2> start interface 1 [ENTER]
```

Case 4: Both local and remote servers are set to *CHAP* authentication protocol.

- Execute *Part Two: Configuring the Local Server*.
- Instruct the specified PPP link at the local server to use *CHAP* authentication and to authenticate users according to an internal table. Execute:

```
Server1> change ppp link 8 authentication chap
auth_protocol table [ENTER]
```

- To start the link at the local server, execute:

```
Server1> start interface 1 [ENTER]
```

- Execute *Part One: Configuring the Remote Server*.
- Set the specified PPP link at the remote server to use *CHAP* authentication and to authenticate users according to an internal table. Execute:

```
Server2> change ppp link 8 authentication chap
auth_protocol table [ENTER]
```

6. To start the link at the remote server, execute:

```
Server2>    start interface 1 [ENTER]
```

Case 5: Both local and remote servers are set to PAP authentication protocol.

1. Execute *Part Two: Configuring the Local Server*.
2. Instruct the specified PPP link at the local server to use PAP authentication and to authenticate users according to an internal table. Execute:

```
Server1>    change ppp link 8 authentication pap  
            auth_protocol table [ENTER]
```

3. To start the link at the local server, execute:

```
Server1>    start interface 1 [ENTER]
```

4. Execute *Part One: Configuring the Remote Server*.

5. Instruct the specified PPP link at the remote server to use PAP authentication and to authenticate users according to an internal table. Execute:

```
Server2>    change ppp link 8 authentication pap  
            auth_protocol table [ENTER]
```

6. To start the link at the remote server, execute:

```
Server2>    start interface 1 [ENTER]
```

Disabling Authentication

Turn *off* all authentication in a server by executing the following command:

```
Local>    change ppp link {link number} authentication none  
            [ENTER]
```

RIP Interface Configuration

This section explains how to configure server ports to use Routing Information Protocol (RIP). An *Overview of RIP* follows the setup procedure. An explanation of many of the command parameters appears in appendix A.

In the RIP link configuration (figure 2-17), Server1 will connect Host1 to Host2 through Server2. The hosts are located on separate Ethernets. Server1, on Host1's Ethernet, is connected via a RIP link to Server2, which is on Host2's Ethernet. Each Server must be assigned its own unique IP address.

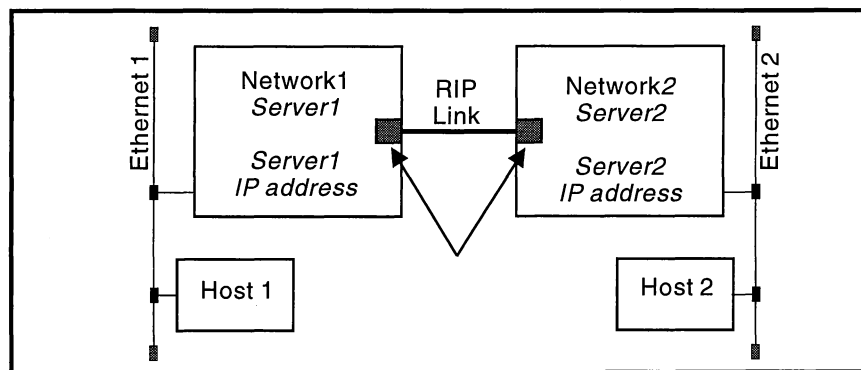


Figure 2-17. Configuring servers for RIP

When you establish a link between local and remote servers, coordinate all aspects of the RIP link configuration with personnel at the remote location. To configure the server for RIP, do the following:

1. Confirm the address of Server1. Execute:

```
Server1> show address [ENTER]
```

 Repeat the above command for Server2.
2. Confirm the route to the Ethernet for Server1. Execute:

```
Server1> show routes [ENTER]
```

 Repeat the above command for Server2.
3. Check the state of the interface at Server1. Execute:

```
Server1> show interface [ENTER]
```

The interface must be UP. If it is DOWN, recheck the SLIP or PPP configurations for abnormalities. Correct the configuration so the interface is UP.

Repeat the above command for Server2.

4. Configure Server1 to use RIP with Server2. Execute:

```
Server1> change rip {rip index number} gateway {Server2
main IP address} interface {type (see table)}
[ENTER]
```

5. Configure Server2 to use RIP with Server1. Execute:

```
Server2> change rip {rip index number} gateway {Server1
IP address} interface {type (see table)} [ENTER]
```

The interface type is determined as follows:

Interface Type Determination	
For RIP via . . .	Use interface type:
SLIP	Serial
PPP	PPP
Ethernet	Ethernet

6. Activate RIP on Server1. To broadcast routing information, use ACTIVE. To listen to and store all routing information, without using it to update other hosts, use PASSIVE. Execute:

```
Server1> change rip enable active [ENTER]
```

Repeat the above command for Server2.

7. Change timeout value, if necessary. If an expected routing update is not received in this time period, it is assumed the gateway is down or that the network to the gateway is faulty. Execute:

```
Server1> change rip timer {timer value} [ENTER]
```

If you do not change the timer value, the server automatically sets it to 30 seconds. Repeat the above command for Server2.

8. Enable or disable Proxy RIP. Execute:

```
Server1> change rip proxy {enable} [ENTER]
```

or

```
Server1> change rip proxy {disable} [ENTER]
```

Repeat the above command for Server2.

RIP

The Internet Routing Information Protocol (RIP) is one of several routing protocols that uses a *vector-distance* algorithm. Distance is interpreted in different ways by different protocols; but, in general, it is a measure of the efficiency of a route to a destination and is referred to generically as a routing *metric*.

RIP's metric is a *hop count*, which is simply the number of gateways a datagram must traverse to reach its destination. Each gateway in the path counts as one hop. If a datagram must pass through one gateway, the hop-count is 1; if it must go through two gateways, the hop count is 2, and so forth. The RIP metric can range in value from 1 through 16, with 16 used to designate a network that is unreachable.

Each gateway using RIP keeps a routing table with an entry for each route. An entry consists of: (1) a destination address, (2) a metric that indicates the distance (number of hops) from this gateway to the destination and (3) the address of the *next-hop* gateway that is, the gateway to which a datagram should be sent next, in order to reach the destination.

A gateway running RIP broadcasts messages every 30 seconds to each of its directly attached networks. Each message lists pairs of destinations and metrics from the senders routing table. These tell a receiving gateway how far away each destination is from the sending gateway. A gateway receiving a RIP routing message does the following for each destination-metric pair in the message:

1. First, the receiving gateway checks its routing table to see if it contains an entry for the destination.
2. If the table does not contain such an entry, the gateway enters the destination and [metric + 1] into the table, with the senders address as the next-hop gateway.
3. Otherwise, the gateway compares the next-hop address and metric value currently in the table with those in the message.

If the metric in the message is lower (better) than the one in the table, the gateway replaces both the next-hop address and the metric in the table with the ones in the message. If the metric in the message is higher (worse) than the one in the table, and the next-hop gateways are the same, the gateway ignores the entry.

If the metric in the message is higher (worse) than the one in the table, and the next-hop gateways are *different*, the gateway ignores both the next hop address and the metric in the message.

To handle gateway and network failures, RIP uses a timeout value of 180 seconds. If an expected routing update is not received from a gateway in 180 seconds, it is assumed that the gateway is down or that the network to it is faulty. For example, if Server A is using Server B as the next hop to destination X, and Server

A has not heard from Server B in 180 seconds, then Server A assumes that it should not route through B to get to X. To ensure that it does not use the invalid route, Server A assigns a metric of 16 (meaning unreachable) to the routing table entry that shows B as the next-hop gateway to X, for a time interval of 120 seconds, during which it will advertise the route. After 120 seconds, it will delete the route.

There are at least two obvious limitations to RIP. One is that it cannot support internetworks in which the longest path involves more than 15 hops. Another is its use of distance alone to measure the desirability of a route. Other factors, such as network congestion and bandwidth, can create significant differences in the desirability of two routes that have equal hop counts. While you can modify RIP distance values to take such factors into account, the protocols designers warn that doing so may cause inconsistencies that the protocol is not designed to handle.

When you use RIP, HELLO and/or EGP in some combination on the same interface, you can specify a preference for routes learned via one protocol versus routes learned by another. In this way, you can adjust the relative impact of each protocol to meet the individual needs of each of your subnets.

Proxy RIP

Proxy RIP solves some of the problems inherent in dialup lines that use “regular” RIP.

A server equipped with activated RIP sends and receives periodic update messages with its connected gateways. The periodic updates help the server maintain its onboard RIP database; that is, keep it filled—often to overflowing—with current routing information.

Thankfully, connections rarely last forever and, if the server fails to receive updates from a particular gateway for a period of time equal to the RIP Gateway Interface Timer value, it removes all traces of that gateway’s existence from its RIP database. The result is passed to any gateways connected to the server via a static link. They, too, soon remove the derelict gateway from their respective RIP databases.

The problem that occurs really is a choice between saving money or saving data. That is, if you drop the dialup link (to save its ongoing cost) whenever all the dialup sessions on that link go away, and the gateway timer times out, you lose the routes to hosts via that gateway. But, if you maintain the link at all times (whatever the cost), you—and the gateways that depend on you—always have a fresh image of the most current routes to target hosts via that gateway.

Maintaining a dialup link only to exchange routing information probably is not cost-effective, and it would be better if the link could be dropped when it isn’t carrying active sessions.

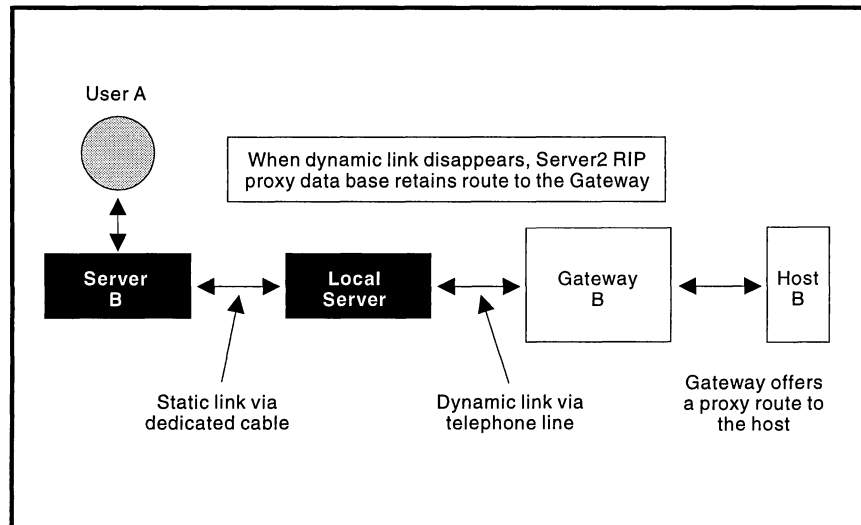


Figure 2-18. Proxy RIP saves money and data

Configuration

How Proxy RIP Works

The server uses Proxy RIP to save money and data (figure 2-18). A server equipped with activated *Proxy RIP* only maintains a telephone link for the duration of user activity on that link. During link uptime, the server also collects routing information for its RIP database. The difference between *standard* and *Proxy RIP* is what happens to the routing data when the link drops and remains down when the RIP gateway interface timer expires. With *standard* RIP, routing data is erased; but with *Proxy RIP*, routing data is maintained.

Aside from the cost savings of dropping the link, another benefit of maintaining proxy routes is the speed with which a user can connect to a remote gateway via a proxy route stored in the intermediary server RIP database. Rather than poll all of its attached gateways for the best route to the target host, and then wait for an answer, the server simply restarts the proxy route via the link to the RIP gateway.

SLIP Configuration

SLIP (Serial Line Internet Protocol) allows Ethernet users or host processors to exchange data with devices that either cannot be connected directly to the Ethernet or that reside on an Ethernet other than the one in which the source user resides. Two types of SLIP configuration are explained: *Static* SLIP (below) and *Dynamic* SLIP. Dynamic SLIP configuration is covered in a subsequent section in this chapter.

In the SLIP line configuration (figure 2-19), the stand-alone Server1 will connect to an Ethernet through Server2. The objective is to configure a serial port for SLIP on each server. Each server is assigned a unique “main” IP address, while

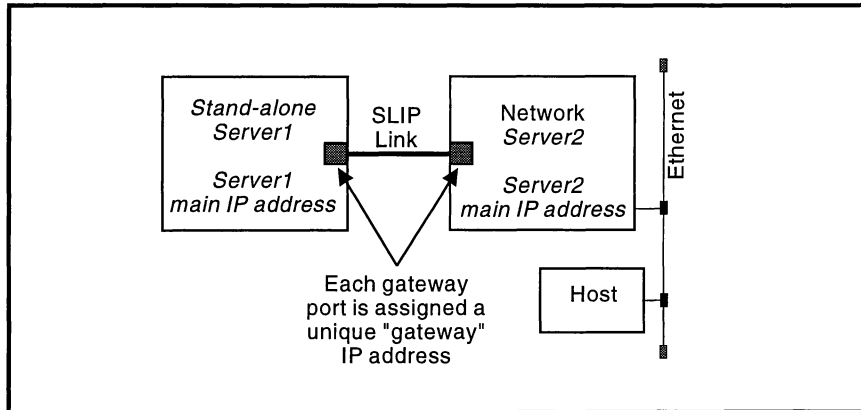


Figure 2-19. Configuring servers for SLIP

the serial ports connecting the servers are gateways through their respective servers. Each gateway port is assigned a unique “gateway” IP address.

When you establish a link between local and remote servers, coordinate all aspects of the SLIP link configuration with personnel at the remote location.

To configure the servers for static SLIP, proceed as follows.

1. At Server1, assign privileges to your port. Execute:

```
Server1> set privileged [ENTER]
```

2. Supply the privileged password. Execute:

```
Password> {privileged password} [ENTER]
```

3. Assign Server1 a main IP address. Execute:

```
Server1> change address {main IP address of Server1}
mask {subnet mask, if applicable} [ENTER]
```

4. Determine available interface numbers at Server1. Execute:

```
Server1> show interfaces [ENTER]
```

5. Select an available interface. Assign a gateway IP address to the interface dedicated to the SLIP link. Execute:

```
Server1> change address {IP address of SLIP interface}
interface {interface number} mask {subnet mask,
if applicable} [ENTER]
```

6. Assign characteristics to the interface port. Execute:

```
Server1>    change port {port number} access dynamic
            autobaud disabled flow control disabled [ENTER]
```

7. Assign the SLIP link to the interface. Execute:

```
Server1>    change interface {interface number} port
            {port number} type serial state up autoinit
            enabled compression disabled [ENTER]
```

8. Determine available route numbers. Execute:

```
Server1>    show routes [ENTER]
```

9. Select an available route number. Create a route to the network through Server2. Execute:

```
Server1>    change route {route number} destination {IP
            address of SLIP interface port at Server2}
            gateway {IP address of SLIP interface port at
            Server1} interface {interface number at Server1}
            mask {subnet mask, if applicable} type
            network direct enable [ENTER]
```

10. Determine the location of Server2. Coordinate the configuration procedure with personnel at the remote location.

11. At Server2, assign privileges to your port. Execute:

```
Server2>    set privileged [ENTER]
```

12. Supply the privileged password. Execute:

```
Password>   {privileged password} [ENTER]
```

13. Assign Server2 a main IP address. Execute:

```
Server2>    change address {main IP address of server2}
            mask {subnet mask, if applicable} [ENTER]
```

14. Determine available interface numbers at Server2. Execute:

```
Server2>    show interfaces [ENTER]
```

15. Select an available interface. Assign a gateway IP address to the interface dedicated to the SLIP link. Execute:

```
Server2>    change address {IP address of SLIP interface}
            interface {interface number} mask {subnet mask,
            if applicable} [ENTER]
```

16. Assign characteristics to the interface port. Execute:

```
Server2>    change port {port number} access dynamic
            autobaud disabled flow control disabled [ENTER]
```

17. Assign the SLIP link to the interface. Execute:

```
Server2>    change interface {interface number} port
            {port number} type serial state up autoinit
            enabled compression disabled [ENTER]
```

18. Determine available route numbers. Execute:

```
Server2>    show routes [ENTER]
```

19. Select an available route number. Create a route to the stand-alone host. Execute:

```
Server2>    change route {route number} destination {IP
            address of SLIP interface port at Server1}
            gateway {IP address of SLIP interface port
            at server2} interface {interface number at
            server2} type host mask {subnet mask, if
            applicable} [ENTER]
```

20. Associate the IP address of the SLIP interface port at Server1 with the Ethernet address of Server2. Execute:

```
Server2>    change arp {IP address of SLIP interface at
            Server1} ethernet {Ethernet address of server2}
            proxy enabled [ENTER]
```

Dynamic SLIP Configuration

The Communications Server supports both static and dynamic SLIP interfaces. Recall that static interfaces remain dedicated (via hard-wire) to the SLIP interface at all times. Server ports that are dedicated to the static SLIP interface cannot be used for other interfaces, such as Ethernet connections.

Dynamic SLIP interfaces become dedicated to SLIP lines on demand. Thus, when a SLIP line is not active at the server port, that port is available for connections to other interfaces. When a SLIP line is needed at the server port, the user executes a short configuration procedure to start the SLIP interface. A useful application for dynamic SLIP is a PC user running software that supports TCP/IP. Such software permits the user to transfer files, exchange mail and act as a Telnet host.

By reconfiguring the line connecting the PC and the server from asynchronous operation to a SLIP link, the server becomes an IP gateway and the PC becomes an IP host. Note that the SLIP line need not be a solid wire; instead, each device can be connected to a modem. See figure 2-20.

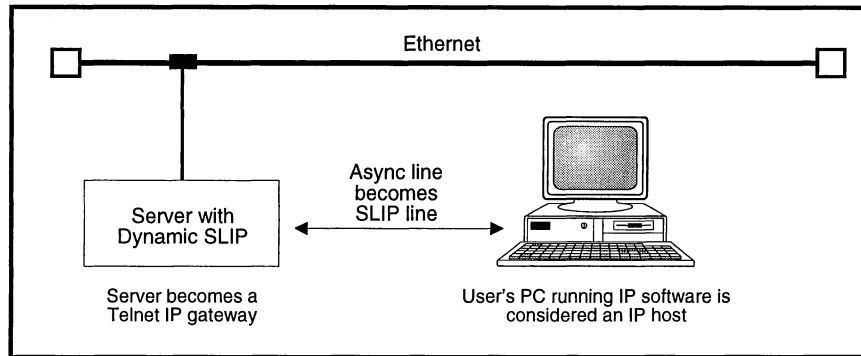


Figure 2-20. Changing async line to SLIP line

To configure a dynamic SLIP line connecting a user's PC with the server, proceed as follows:

1. The PC must contain a terminal emulation program, such as *HBScorn*, which permits the PC to access the server. When that application is loaded into the PC, start the program and connect to the server port.
2. Upon connecting, awaken the target server port and obtain the Local> prompt. Execute:

[ENTER] [ENTER]

The local server prompt appears.

Local>

3. Confirm that the following port parameters are activated:

- Port Access Dynamic
- Port Flow Control Disabled
- Port Autobaud Disabled

Execute:

Local> **show port characteristics [ENTER]**

4. Change the port parameters to the required settings, if necessary. Execute:

Local> **set port access dynamic flow control disabled autobaud disabled [ENTER]**

5. Now the server port is ready for SLIP. Start the SLIP connection by calling the IP address of the PC from the server port. Execute:

Local> **connect slip {IP address of user's PC} [ENTER]**

6. When the connection is established, a message appears at the user's PC:

Session Established

7. Exit from the terminal emulation program and return to the PC DOS prompt.

Note: The syntax of the **exit** command or the assigned **EXIT** function key depends upon the terminal emulation package installed in the PC. Check your application user documentation for further information.

For example, if you are using the HBScom terminal emulation program, execute:

HBScom> **exit** [ENTER]

8. The PC screen should display the DOS prompt.

C:\>

9. Start the the PC-resident Telnet application software program which configures the PC as a Telnet host. Execute:

C:\Local> {start command} [ENTER]

START COMMAND—The character string which activates the PC-resident Telnet application software. For example, if **net** is the command used to start the application, execute:

C:\Local> **net** [ENTER]

10. At the application prompt, connect to the desired destination host using either the IP address of the host or the name of the host (by query to the domain nameserver):

PROMPTLocal> **connect Telnet** {IP address of destination
host} [ENTER]

or

PROMPTLocal> **connect Telnet** {name of host} [ENTER]

11. To end the Telnet session, break out of the host and return to the application prompt.

HOST PROMPT> [BREAK]

12. You are returned to the local prompt of the server. Either continue or terminate the session. Execute:

Local> **disconnect** {number of session} [ENTER]

Compressed SLIP

Compressed SLIP is a recent development in TCP/IP communications.

The strategy of compressed SLIP is to improve data throughput on SLIP lines. Throughput improvement is achieved by shortening the TCP/IP header, which is part of every packet. Recall that during the typical interactive Telnet session only one byte of data is sent across the internet. However, the TCP and IP headers required to send that single byte, together consume at least 40 bytes. When a serial link is used to exchange the data packets and acknowledgments, two packets are typically required for that single byte of data. First, there is the data packet containing 41 bytes, and then, there is a data acknowledgment packet containing 40 bytes, which returns from the destination host to the source.

The resulting traffic ties up the serial link, reducing the typing speed bandwidth that should be available given the speed of the link. Unlike a hardwired, 9600 baud connection, in which the total 9600 bytes are available for data transfer, here only $\frac{1}{40}$ th of the total capacity of a hardwired 9600 baud connection is available.

Experts estimate the value $\frac{1}{40}$ th because the TCP/IP connection is a full duplex connection. In such a connection, the returning acknowledgment packet can pass the next outgoing new packet. Obviously, $\frac{1}{40}$ th of 9600 baud isn't very fast, especially when you realize that the objective is to transfer a single byte of data across the link and that 40 bytes of header are consumed in that process.

In SLIP the problem is magnified because additional characters control the frame *startup*, *stop* and *escape* functions. Note that this same problem occurs in other protocols, such as *PPP*. Therefore, the header compression solution used in SLIP links applies equally to other link level protocols, such as *PPP*. TCP/IP packet headers are compressed using a technique called the *Van Jacobson header compression algorithm*. The same algorithm used to compress the packet headers is also used to expand them upon arrival at the destination host.

The server implements this compression algorithm as compressed SLIP, which uses the Van Jacobson header compression algorithm with the SLIP extensions as described in RFC1144.

Compression in a SLIP interface can be Active, Enabled or Disabled:

- If you set compression to *Active*, all TCP packets compressible according to the RFC1144 specification are sent compressed. In this mode, the server can receive all normal TCP/IP packets, and compressed or non-compressed headers. Typically, 80-95% of headers are compressed down from 40 bytes to between 3 and 5 bytes.
- If you set compression to *Enabled*, the server doesn't send any compressed packet headers, but it can recognize them. Upon detecting incoming compressed headers, outgoing packet headers are compressed.

- If you set compression to Disabled, the server will not attempt to send compressed packet headers regardless of the contents of the packets it receives. This setting permits users to eliminate compression from the data transmission process, which is useful for debugging a link. To set a serial interface for compressed SLIP, execute:

```
Local>  change interface {number of interface} type serial port
        {number of port} state up compression {compression
        parameter} [ENTER]
```

TCP/IP

The Internet Protocol (IP) part of TCP/IP provides a mechanism for transporting data packets between different machines on either a LAN or a WAN. The Transmission Control Protocol (TCP) part of TCP/IP guarantees that data packets will travel dependably from an exact location on one machine to an exact location on another machine.

While TCP works without demanding much of the network manager, IP does its job only if the network manager correctly assigns an acceptable IP address to each host machine on the local network. Provision must also be made to communicate with remote networks that lie beyond the local network. Thus, it is important to understand how to set up IP addresses and employ IP routing.

In essence, when a packet leaves a host machine, it bears the IP address of the source host as well as the IP address of the target machine. If both the source and target machines are on the local network, the target machine will scoop up the packet. However, if the target machine is on a remote network, the packet must be routed to the target network and then to the target machine. If the target machine is up and running, it will capture the packet.

IP Addresses

Each IP address is a unique string of 32 bits and contains identifiers for both the network on which a machine resides and for the machine (host) itself. There are five classes of addresses, although only classes *A*, *B* and *C* are of interest to the majority of network users. IP Addresses are formatted into three parts: a network type identifier (or, address class), a network ID and a host ID. The difference between the classes is the number of bits assigned to each part of the address.

For example, the zero bit (which really is the *first* bit) of a class *A* address is set to zero. Bits one through seven identify the network, and bits eight through 31 identify host machines. In a class *B* address, the first two bits are 1 and 0. Bits two through 15 identify the network. Bits 16 through 31 identify host machines.

For convenience, IP addresses are written as four decimal integers separated by dots. Each integer equals the value of one octet of the IP address. Thus, the binary string: 11000000 01100000 00110000 00000111 can be simplified to: 192.96.48.7.

Domain Names

Users may find it difficult to remember long strings of what appear to be meaningless digits. The resulting domain naming scheme, which associates a series of one or more alpha character name strings to an IP address, further simplifies the process of connecting to Telnet hosts. To make the domain naming scheme work, one or more domain nameservers are located on every local network. The domain nameservers link domain names to the IP addresses needed to route packets.

Domain Nameservers

When a server user executes a **connect** command to a domain name, the local server looks in its nameserver table for the IP address of the target host. If it cannot find the domain name in its own table, it dispatches a request to the external nameserver that serves that domain. The remote nameserver returns the IP address of the target host and the local server can then begin exchanging packets with the target host.

Access to a nameserver can be added so that host machine names can be resolved into IP addresses for those machines. A nameserver is strictly involved in resolving domain names to IP addresses. Think of a nameserver as something like a telephone book in which people's names are replaced with host domain names and the telephone numbers are replaced with IP addresses. Note that a nameserver does not route the packets.

Routers

A router accepts packets that are sent to it and assumes responsibility for getting the packets to their destinations. The router's table of routes acts as a roadmap showing paths to host IP addresses.

For each IP packet that arrives at the local server, the server looks at the network portion of the packet's target IP address and compares that with the network portion of its own IP address. If the network portions match, the local server takes responsibility for delivering the packet to the target host.

However, if a packet has a target IP address that is not on the server's physical network, the server is not responsible for delivering that packet to the target host. Instead, the server checks its local routing table for a target to which it can send the packet.

If the target IP address is listed, the packet is dispatched. However, if the IP address is not listed, and it is not on the server's physical network, the server's last resort is to dispatch the packet to a default route. It then becomes the responsibility of the default router to get the packet to its destination host.

Assigning an IP Address to Your Server

The local server must have an IP address before you can connect to a Telnet host. You can assign a new IP address using the parser commands outlined in the procedure below, or you can use the configurator to assign a new IP address. If the server already has an IP address and you want to change that IP address to a new IP address, you *must* use the parser commands.

1. Log into the server.
2. Set the port through which you will be operating to *Privileged* status. Execute:

```
Local>      set privileged [ENTER]
```

3. You are prompted for a Privileged password.

```
Password>   {privileged password} [ENTER]
```

Note: Optional: Execute the **clear events** command to clear the Events Log. If you clear the Events Log *before* starting your sessions, should any errors occur during your sessions, the Events Log only will record the latest error messages and no others.

To clear the log, execute:

```
Local>      clear events [ENTER]
```

5. Assign a Telnet IP address to the server.

Note: You can assign more than one IP address to the server.

Execute:

```
Local>      change address {IP address of server} ENTER
```

The IP Address is a string of digits in the form *i.i.i.i*, where *i* is an integer from 0-255.

Adding Access to a Nameserver

During operation, when a user wishes to connect to a Telnet host name or IP address that is not within the local cache, the server queries an external nameserver to find the desired host address within that nameserver. If the first nameserver cannot resolve the host name address, that nameserver contacts other nameservers on the network in an effort to locate the desired host. If the route can be determined and the connection is completed, the user sees the login prompt for the destination host. However, if no route is determined, the user probably will see the message, Host or Service {IP address} not known.

The server automatically fills an internal table with names of system hosts it has resolved using an external nameserver. Access to these systems (on subsequent

attempts) will be faster. However, these entries only stay in the server's memory for a period limited by the TCP/IP settings of those systems.

1. To enable the server to access a primary nameserver, from which it can obtain IP addresses corresponding to names of other hosts, execute:

```
Local>  change nameserver {IP address of host known to be
        nameserver} [ENTER]
```

Note: The host nameserver need not be on the same local network as the server.

Creating a Route to a Gateway

If server users will be connecting to hosts beyond the local network, create a route between the server and a gateway that will route packets from a source network to receiving gateways on destination networks.

1. Each route is identified by a unique ordinal number. To determine which ordinal numbers already are in use, generate a display of the currently active routes. Execute:

```
Local>  show routes [ENTER]
```

2. To create a route to a gateway, execute:

```
Local>  change route {unused route number, equal to or less
        than 32} gateway {IP address of gateway} type default
[ENTER]
```

3. To check that the new route has been added to the server local cache, execute:

```
Local>  show routes [ENTER]
```

Creating a Local Telnet Host

Create local hosts that accept incoming (reverse Telnet) connects from remote users.

1. To create a local Telnet host that will be available through the server, execute:

```
Local>  change service {name of Telnet host} port {number of
        physical server port} tcp port {number of TCP port}
        Telnet enable lat disable identification
        "{identifying string}" [ENTER]
```

2. Users can connect to this host by connecting to the server's IP address with the destination socket (TCP port) designated according to the format:

```
Local>  connect Telnet {name of host} destination {number of
        TCP port}
```

3. For each physical server port supporting connections to the local host, execute:

```
Local>  change port {number of physical server port} access
        dynamic autobaud disabled [ENTER]
```

Setting Host Names/IP Address in Local Cache

Users find it easier to remember recognizable words or names than a string of unrelated digits. Associating a host name with an IP address is an attempt to simplify the connection procedure.

Recall that the domain name is a sequence of short, alpha character strings separated by dots. A complete domain is written from lowest level domain first, to the highest level domain last.

For example, *DOC.DSS.COM* is a domain name that identifies the service *Doc*, at Penril Datability Networks, Inc., which is a commercial organization.

The server's local cache contains IP addresses and respective domain names. These names are populated manually or automatically by the server manager through its experience of completing connections to Telnet hosts.

1. To manually populate the server's local cache, execute:

```
Local>  change domain {name of Telnet host} IP {IP
        address of Telnet host} [ENTER]
```

2. Users can log into the server and connect to Telnet hosts either by furnishing the host's IP address or by entering the pre-determined host name, (or domain) which the nameserver can resolve. Test the new entry either by connecting to the host IP address or to the host name. Execute:

```
Local>  connect Telnet {IP address of Telnet host} [ENTER]
```

```
Local>  connect Telnet {name of Telnet host} [ENTER]
```

Note: If you issue the command, **Show NameServer**, you can see the local list of host names and associated IP addresses that reside in the server.

Rlogin

Rlogin protocol is used primarily to log into a UNIX-based Telnet host. It simplifies the login process by allowing a user to connect to a network host *without* the additional steps of entering a username and password. Rlogin also passes the user's terminal type (a character string in the user's port profile which tells the host how to communicate with the user's terminal) across the network to the remote host.

If you are logged into the server, and you issue a **connect rlogin** command, the server sends your port username to the destination UNIX host. The destina-

tion host checks its Rlogin database for your port username. If your port username is found, you are connected to the host and can begin executing commands from the host prompt. If your port username is not found, you are prompted for a valid login password before you can begin using the host.

Adding users to the Rlogin database is a UNIX system administrator's function, which is beyond the scope of this document. For further instructions, consult your UNIX system administrator's documentation.

Note: Server Rlogin implementation only supports connections to the remote host (forward Rlogin); it does not accept connections to the server from the remote host. To log into a UNIX-based Telnet host with Rlogin, proceed as follows:

1. Log into the local server. When you enter your username, that string is sent to the target system. Note that the target system may discriminate between upper- and lower-case characters. Enter your username using the correct case. For example:

```
Enter username> McNab [ENTER]
```

2. When you execute the **connect rlogin** command, you can select either a destination IP address or a destination Telnet hostname.

Execute:

```
Local> connect rlogin {destination IP address} [ENTER]
```

or

```
Local> connect rlogin {name of destination host} [ENTER]
```

3. Your username is forwarded to the target UNIX host. If your port username is not known to the UNIX host, you are prompted for a host login password. If you do not know this password, contact the UNIX system administrator for further information.

```
Password>
```

Connecting to the TCP Remote Console

The TCP remote console feature allows a remote user to connect to the server to execute management or configuration commands. Thus, by logging into the remote console, a network administrator, seated at a single location, can manage any server on the network.

You can connect from any TCP/IP device on the Ethernet to a server if that server has been assigned an IP address. When you assign a Telnet IP address to the server, the server console port automatically is set to port *2048*.

You can connect to this port if you specify the console port number when you connect to the IP address. The connection is password-protected to limit access only to authorized users.

1. To connect to the remote console, execute:

```
Local> connect {IP address of target} destination 2048 [ENTER]
```

or

```
Local> connect {IP address of target}/2048 [ENTER]
```

2. When the connection is established, you are prompted with:

```
Local> -nnn- Session n to {IP address}/2048 established
```

3. Press **ENTER** and you are prompted with a #.

```
#
```

4. Enter the password at the prompt and you are prompted with the normal server welcome message. The default remote console password is *ACCESS*. Your connection acts just like a directly connected terminal until you terminate it.

Notes: You cannot execute file transfers over the remote console.

You can connect to network services; however, you can maintain only one session.

You must use the tilde (~) to switch between local mode and service mode. (The ~ serves as a **BREAK** key when you are using RCF.)

Telnet Security

Telnet security controls access to and from the server. Use the Telnet security feature to limit:

- *Incoming* connections from remote users to Telnet hosts attached to ports in the server
- *Outgoing* connections from users at server ports to destination IP addresses.

When Server Ipsecure is disabled, incoming connections to the server are accepted and are *not* screened through Telnet security. However, when Server Ipsecure is enabled, all incoming connections to the server are screened through Telnet security to determine if they can be accepted. In a similar fashion, Telnet security controls all outgoing calls from the server to destination hosts.

How Telnet Security Works

Server Line Card ports can be attached to local devices offered as Telnet hosts. Network users can start sessions with these hosts by connecting to either the IP address or the Telnet name of the server. Similarly, local server users can execute requests to connect to remote hosts.

A Telnet module resides within the server. When Server Ipsecure is enabled, the Telnet module screens every incoming connection to the server from a Telnet host, and every outgoing connection from a server port to a Telnet host.

For incoming connections, the screening process begins when the Telnet module refers to the server database to determine the server Telnet group numbers. The server is assigned its group numbers to control incoming access *from* Telnet hosts, just as each server port is assigned its group numbers to control outgoing access *to* Telnet hosts.

Recall that a security structure is a relationship between one or more Telnet group numbers, plus an address mask and a security address. The entire structure is identified by a security ID name. These structures are created by the server manager. Up to 32 security structures can reside in the server database.

When incoming connections are received, the Telnet module compares the *security structure* group numbers with the *server* group numbers. Each time a security structure group number matches a server group number, the Telnet module executes a test using the contents of the corresponding security structure and the IP address of the host that sent the connection request.

The test is a logical *AND* operation that combs the IP address of the source host through the address mask of the security structure. The result is compared to the security address of the security structure. If the result matches the security address, the IP address passes the test.

If the IP address passes the test, the incoming connection is accepted to the destination host, which is connected to a port in the server. If the IP address fails the test, the module continues to compare group numbers until either another match is found (and another test is administered), or no other match is found and the connection request is rejected.

Outgoing connections from the server to a Connections from Server to Telnet host are tested in a similar fashion except that:

- The server *port* group numbers are compared with security structure group numbers.
- The *destination host IP address* is combed through the address mask of the security structure. Then, the result is compared to the security address of the security structure.

Telnet Security Masks

Recall that an IP address contains both network- and host-identifier bytes in ratios determined by the class (A, B or C) of the Telnet network in which the server is installed (e.g., class B: *net byte.net byte.host byte.host byte*).

An IP address mask is a string of bytes in the format, *n1 . n2 . n3 . n4*. The address mask is logically *AND*ed to the IP address of the source host. The result of the logical *AND* operation is a string of significant bits. The significant bits often are the network portion of the IP address; however, an IP mask can be designed to pass any of the bits in an IP address.

For example, consider a mask that only looks at the first 20 bits of a class B IP address: 11111111 . 11111111 . 11110000 . 00000000. Converted to its decimal format, the mask is 255 . 255 . 240 . 0. The first 20 bits of the IP address define the two network bytes and a host byte. The remaining host bits are ignored.

The resulting string of bits is compared to a predefined security address. All of the significant bits must match or security clearance is denied. A security mask is illustrated in figure 2-21.

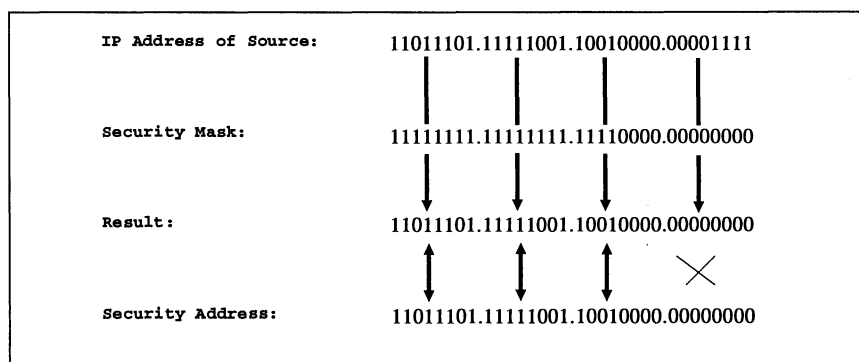


Figure 2-21. Logically *AND*ing address and mask

Telnet Security Addresses

A Telnet security address is a string of bytes in the format, *n1.n2.n3.n4*.

Telnet security screening is a two-part process. After the Telnet module passes the significant bits of the source IP address through the security mask to create a result string, it compares each bit of the result string with its corresponding bit in the security address.

Each bit of a security address is carefully selected to correspond to the bits in the source IP address that will pass through the security mask. For example, observe the security structure illustrated in figure 2-22. To obtain access to the server, the first three bytes of the host IP address must be 221.249.144. The fourth byte is not significant and can be any value. In this example, incoming connections to the server are accepted from host IP addresses 221.249.144.0 to 221.249.144.255.

Thus, to create security addresses, one must know the bit content of the expected source IP addresses. Then, one must design a security mask to pass the significant bits of the expected IP addresses so they will match the bits of the security address.

Look for common network or host bytes in the IP addresses of acceptable clients. Then, set up a structure with a security mask that only passes those bytes to the result. The appropriate security address only would match the bits in those bytes, allowing client access to the server, as illustrated in figure 2-22.

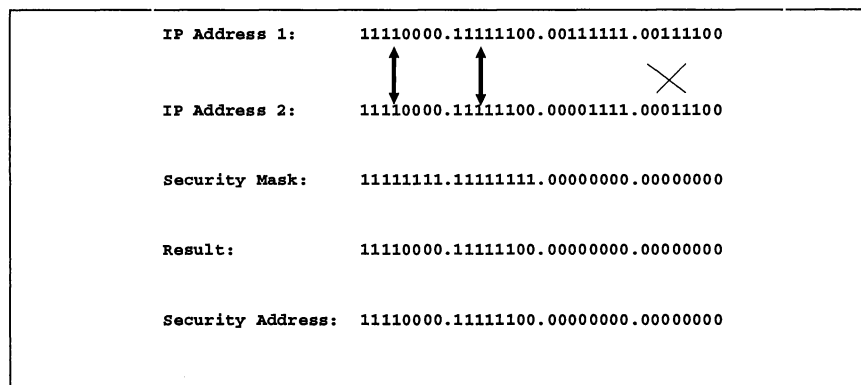


Figure 2-22. Pass common bytes to security address

As more source IP addresses are added to the list of acceptable clients, it becomes necessary to create many security structures, each containing a unique security mask and security address and a group number that matches one of the server group numbers.

Recall that the server checks only those security structures that have group numbers in common with its own group numbers. Common group numbers

ensure that when the server receives an incoming connection request, the Telnet module will find a security structure to test the IP address against.

For outgoing calls to remote hosts, the security mask and security address operate on the IP address of the destination host. The security structure group number must match the *port group* number, or the Telnet module will not find a security structure to test the outgoing call, and the server will deny the connection request.

Note: To simplify the security structure set up procedure consider using an indiscriminate mask, 255.255.255.255, which passes all the digits of the source host IP address. Then, simply set the security address to the single IP address that should be permitted access to the server. In that security structure, no other IP addresses will have access to the server.

Or, set the security mask to pass only the network bytes of the source IP address, and set up a security address that only matches those network bytes. The remaining bytes in the source IP address describe the source host. Since the mask does not pass those bytes to the result, they are not matched to the security address. In that way, any host in that source network can connect to the server.

Setting up Basic Incoming Telnet Security

1. Display all of the current security identification names. Execute:

```
Local> show security [ENTER]
```

2. Select an available security identification name. To choose an inactive security ID and activate it, execute:

```
Local> change security {security_ID name} [ENTER]
```

The *security ID name* is a number from 1 to 32, which identifies the Telnet security structure.

3. Create an address mask and assign it to the activated security identification name. Execute:

```
Local> change security {security_ID name} mask {address mask value} [ENTER]
```

The *address mask value* is a string of numbers in the format, *n1.n2.n3.n4*, which specifies the significant bytes. For example, a mask that only looks at the first 24 bits is: 11111111.11111111.11111111.00000000.

Converted to its decimal format, the mask is 255.255.255.0.

4. Select a security address. The format of the security address is *n1.n2.n3.n4*, where *n* is a decimal value from zero (0) to 255. To add the new security address to the security structure, execute:

```
Local> change security {security_ID name} address {security
address value} [ENTER]
```

The *security address value* is a string of numbers in the format *n1.n2.n3.n4*, which specify the security address for the named security structure.

For example, if the security ID name is 1 and the address is 192.41.217.0, execute:

```
Local> change security 1 address 192.41.217.0 [ENTER]
```

5. Assign the security structure to a group number. Execute the following command:

```
Local> change security {security_ID name} group {number(s) of
group(s)} [ENTER]
```

The *numbers of groups* is a list or range of numerals specifying active Telnet group names.

6. Authorize the user's port to use the security groups assigned in step 5. Execute:

```
Local> change port authorized groups {number(s) of group(s)}
enabled [ENTER]
```

7. Assign the same group numbers (from step 5) to the user's Telnet server port. Execute:

```
Local> change port {number of user's server port} group
{number(s) of group(s)} [ENTER]
```

8. Enable the IP security feature for the server. Execute:

```
Local> change server ipsecure enabled [ENTER]
```

Setting up Basic Outgoing Telnet Security

Following is a procedure to set up an IP security structure that restricts a server port user only to IP host addresses whose *first three bytes* are: 192.41.217.

1. Activate the IP security structure controls. Execute:

```
Local> change server ipsecure enabled [ENTER]
```

2. Check the server display for the applicable service group (zero by default).

```
Local> show server [ENTER]
```

3. Create the IP security structure according to the desired parameters. Execute:

```
Local>  change security 1 mask 255.255.255.0 address  
        192.41.217.0 group 0 [ENTER]
```

The GROUP 0 parameter is the number of the port group to which this security structure belongs. A server port can belong to several groups, and therefore, to several security structures. Thus, if the user's request does not pass the scrutiny of the security structure associated with the first port group, the Telnet module repeats the test using each of the remaining security structure-port group relationships.

For information on other command parameters, see appendix A.

Notes: If you do not include the security IP address, by default, the server assigns the illegal address, 0 . 0 . 0 . 0, and the user cannot connect to any network host. If you do not assign a group number in this step, by default there will be no groups, and the user will not be able to connect to any network host.

4. Check that the proper parameters have been selected and assigned to the security structure. Execute:

```
Local>  show security [ENTER]
```

TN3270 Protocol Configuration

This section explains how to use the TN3270 protocol feature to connect an ENIC^{Plus} equipped server via Telnet to an IBM host.

TN3270 is an IBM protocol, developed to allow users of TCP/IP to connect to IBM's proprietary SNA protocol.

The TN3270 capability of the ENIC^{Plus} lets you use a VT terminal to connect via a Telnet link to an IBM host, and then to a 3270 application. To do this, you create a TN3270 profile within the server. This profile identifies both the TN3270 terminal you are trying to reach (for example, an IBM-3278-2) and the local VT device connected to the server (for example, a VT100 terminal). This profile also lets the IBM host know that it will be communicating over a Telnet link to a TN3270 application, as opposed to its communicating directly to a 3270 application (for example, to a 3270 line card in the Communications Server). TN3270 is essentially a bridge between a server and an IBM host. (See figure 2-23.)

The TN3270 software does the work of both the 3270 application and the 3270 gateway. On the server side, a VT device (for example, a VT100 terminal) and the server pass ASCII characters back and forth. Within the server, the TN3270 software converts the ASCII to EBCDIC format, and then to TN3270 format. The TN3270 data is sent over the Telnet link to the TN3270 gateway, which in turn passes it to IBM host and then to the 3270 application. The inverse occurs when data is sent from the remote 3270 application: data in 3270 format is sent from the 3270 gateway, over the Telnet link to the Telnet gateway on the server side.

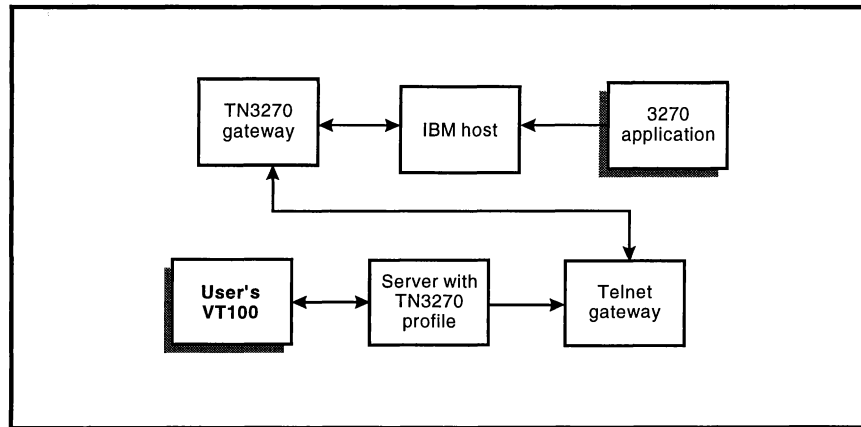


Figure 2-23. Generic TN3270 network diagram

The server converts the TN3270 data first to EBCDIC format, and then to the VT format specified in the server TN3270 profile.

When connecting to the IBM host, the server first makes a TCP/IP connection. Then, a Telnet connection is established. The TN3270 profile in the server then establishes TN3270 communication with the IBM host. The Communications Server supports up to 48 simultaneous TN3270 sessions. To configure the ENIC^{Plus}-equipped communication Server for TN3270:

1. Create a port profile in the server database to use with TN3270. Execute:

```
Prompt> change profile {profile name} [ENTER]
```

Where:

PROFILE NAME—Is an alphanumeric string of up to 16 characters.

2. Specify the 3270 terminal type to connect to for this TN3270 session. Execute:

```
Local> change profile {profile name} terminal {3270 terminal type} [ENTER]
```

Where:

3270 TERMINAL TYPE—The type of terminal this profile will use. Valid TN3270 terminal types are IBM-3278-2, IBM-3278-3, IBM-3278-4 and IBM-3278-5. Enter the type exactly as shown here with the hyphens. If none of these four terminal types is entered, a “normal” (non-3270) Telnet session is assumed.

3. Specify the type of VT terminal you are using. The default is a VT100 terminal. Execute:

```
Local> change profile {profile name} vt_emul {vt terminal
type} [ENTER]
```

Where:

VT TERMINAL TYPE—Type of VT terminal connected to the port on the server Line Card. Choices are VT100, VT220, VT320, VT330, VT340 and VT420.

Keyboard Mapping Profiles (KMPs)

To emulate a TN3270 device, the VT terminal you are using must be able to send data just as though it were an actual 3270-type terminal. While many keys are identical on both types of terminals (such as alpha characters, numerals and symbols), some keys (such as the **PF7** key) only exist at the 3270 terminal and not at the VT terminal. Thus, the additional 3270 keyboard functions must be added to the VT terminal.

To fully duplicate the functions of a 3270 keyboard, the VT terminal interfaces through a Keyboard Mapping Profile (KMP) in the local server. The KMP is a database in which each VT terminal key is associated to its equivalent key on an IBM 3270-type terminal. Six default KMPs are supplied with the server. Any one of these KMPs can be activated within the user's port profile and used as-is, or a default KMP can be copied to a new file name, modified with different keyboard settings and used in the port profile.

Add a required 3270 keyboard function to a VT keyboard by assigning the control sequence of the required 3270 key to an unused VT key. For example, the 3270 keyboard has a **PF7** key, a VT keyboards does not. Add the **PF7** key to a VT keyboard: Access the port profile's KMP and locate an unused key that you can map to the function on the 3270 device. Do not modify the current KMP; create a new KMP and assign the unused VT key the same control sequence as the 3270's **PF7** key.

Two default KMPs exist in the server. Up to eight additional customized KMPs can be added to the server. To create custom KMPs, proceed as follows:

1. A default KMP is automatically is assigned to your VT terminal:

Terminal Type	Default KMP
VT100	VT100
VT220	VT220
VT320	VT220
VT340	VT220
VT340	VT220
VT420	VT220

When you want to create a customized KMP, you create a new KMP name and the server automatically copies the default VT100 KMP into that new file. Then, you modify the copy of the VT100 KMP one key at a time. To create a new KMP, execute:

```
Local>      change kmp {new KMP name} [ENTER]
```

- Existing KMPs *cannot* be modified. If you want to modify an existing KMP and reuse the KMP name, purge the old KMP *before* creating the new KMP. Default KMPs cannot be purged. To remove an existing KMP name, execute:

```
Local>      purge kmp {old kmp name} [ENTER]
```

Note: You can copy an existing KMP to a new name in the operational database. You can modify that new KMP and assign it to a port profile.

Refer to the section on *Copying KMPs*.

- Modify the new KMP *one key at a time*. (For example, to modify five keys, execute the **change kmp** command five times.) To change key assignments, execute:

```
Local>      change kmp {new kmp name} {3270 key} equal "{VT key}"
[ENTER]
```

Where:

NEW KMP NAME—Identifies KMP to be changed.

3270 KEY—Identifies the 3270 key function. Obtain this from the *3270 Key Label* column in *appendix B*.

VT KEY—Identifies the VT terminal keystroke that will execute the 3270 key function. Enclose the key sequence in quotation marks, as shown. (Default VT keystrokes are listed in the *VT Keystroke Sequence* column of *appendix B*.)

- View the newly-created KMP, if desired. Execute:

```
Local>      show kmp {kmp name} [ENTER]
```

Only the first 64 keys in the profile are displayed as these are the only keys that can be reassigned. Keys greater than #64 cannot be reassigned.

- Assign the new KMP to a port profile. Execute:

```
Local>      change profile {port profile name} kmp {kmp name} [ENTER]
```

- Assign the port profile containing the new KMP to a server port. Execute:

```
Local>      change port {target port number} profile {profile name}
[ENTER]
```

7. Log out the server port to activate the new settings. Execute:

```
Local>  logout port {target port number} [ENTER]
```

8. Log into the target port and connect (via Telnet) to the IP address of the target IBM host. When you are logged in, your VT terminal will communicate with the target 3270 application. Execute:

```
Local>  connect {IP address of target host} [ENTER]
```

Copying KMPs

Copying an existing KMP is a convenient way to create a new KMP without starting from *scratch*. For example, if you have created a customized KMP that you wish to use—with just a few modifications—in another port profile, you can copy that KMP to a new name. Once you have copied the KMP to a new name, you can customize it by modifying keyboard functions in the new KMP. Then, you can store that KMP in either of the server's operational or permanent databases. Later, it can be recalled into a port profile and used for TN3270 interactions. If you want to retain the new KMP for long-term use, copy it into the server's permanent database. If the changes only are needed for the duration of the current user session, copy the KMP into the server's operational database.

Note the difference between the **change kmp** command and the **copy kmp** command. The **change kmp** command only copies the *VT100 KMP* into a file; you cannot copy any other KMP using the **change kmp** command. To copy a KMP other than the VT100 KMP, you must use the **copy kmp** command.

Following is a procedure to copy an existing KMP from one port profile to another and to modify the new KMP and retain it for long-term use.

1. Copy the current KMP name to a new name in the permanent database. Execute:

```
Local>  copy kmp {current KMP name} to permanent {new KMP name}
[ENTER]
```

2. Make any desired changes to the new KMP. Execute the following command as many times as needed:

```
Local>  change kmp {new KMP name} {3270 key} equal "{VT key}"
[ENTER]
```

3. Create a new port profile for the new KMP. Execute:

```
Local>  change profile {new profile name} [ENTER]
```

4. Load the new KMP into the new profile. Execute:

```
Local>  change profile {new profile name} kmp {new KMP name}
[ENTER]
```

5. Assign the new profile to server ports, as required.

```
Local> change port {number} profile {profile name} [ENTER]
```

NPT Card for Network Protocol Translation

The Communications Server accepts a line card that supports Network Protocol Translation (NPT). A network protocol translator allows one type of data communications protocol, such as LAT, to be translated to another protocol, such as TCP/IP. The need for protocol translation arises when, for example, a user logs into a LAT-based server and requests to connect to a TCP/IP-based host.

To use the NPT Line Card, your server must be equipped with a dual protocol (LAT – TCP/IP) ENIC. To identify the ENIC in your server chassis, execute the **Show Server Summary** command. The output display contains information about the ENIC. Verify that both the TCP/IP and LAT protocols are supported by the ENIC. If not, the ENIC can be upgraded via password keys. for more information, refer to *Upgrading Server Software via Password Keys* in chapter 4.

The NPT card contains pairs of logical ports. Logical ports are not physical server ports; that is, they are not visible to the user. Unlike physical ports, which are receptacles mounted on the server chassis to accept data cables, logical ports only exist within the NPT software. Note that the backplate of the NPT card (figure 2-24) contains no receptacles.

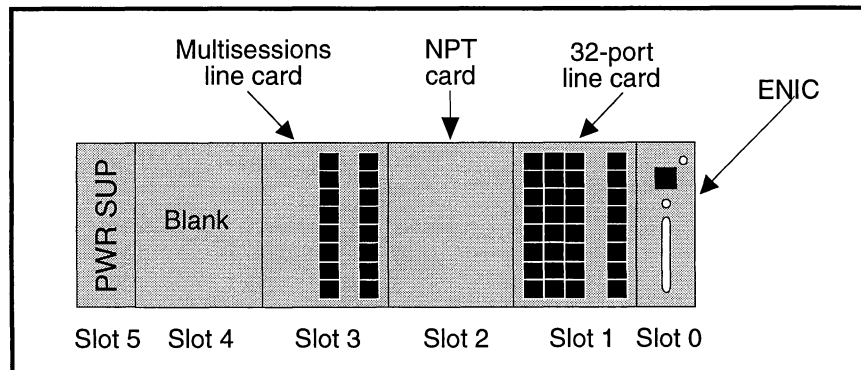


Figure 2-24. NPT configuration with other cards

There are two types of logical ports; input and output. The relationship is as follows:

- Logical input ports receive connection requests from users via the network. NPT software translates the incoming protocol to the protocol required to communicate with the target host
- Logical output ports dispatch the translated packets to the target host. See table 2-1.

Table 2-1. Allocation of port numbers for a Communications Server with sample configuration

Slot 1	Slot 2	Slot 3
<i>32-port card</i> physical ports	<i>NPT card</i> logical ports	<i>Multisessions line card</i> Physical ports Logical Ports
1—32	IN—OUT 33—49 34—50 35—51 36—52 37—53 38—54 39—55 40—56 41—57 42—58 43—59 44—60 45—61 46—62 47—63 48—64	IN—OUT 65—81 66—82 67—83 68—84 69—85 70—86 71—87 72—88 73—89 74—90 75—91 76—92 77—93 78—94 79—95 80—96

To configure for NPT, an input port is assigned a service name to which network users can connect. This service name identifies the target host. In addition to its service name, the input port also is configured with a virtual text string. The virtual text string is read by NPT software, which instructs the server software to execute a connection to the target host. For example, if the virtual string is either the IP address or the domain name of the target host, the server automatically executes a connection to the host identified by the string. When the user connects to the local service name, the NPT software completes the connection to the target host—transparently to the user. Refer to the examples contained in the following pages for more information about Network NPT configurations.

Providing LAT Access to TCP/IP Telnet Hosts

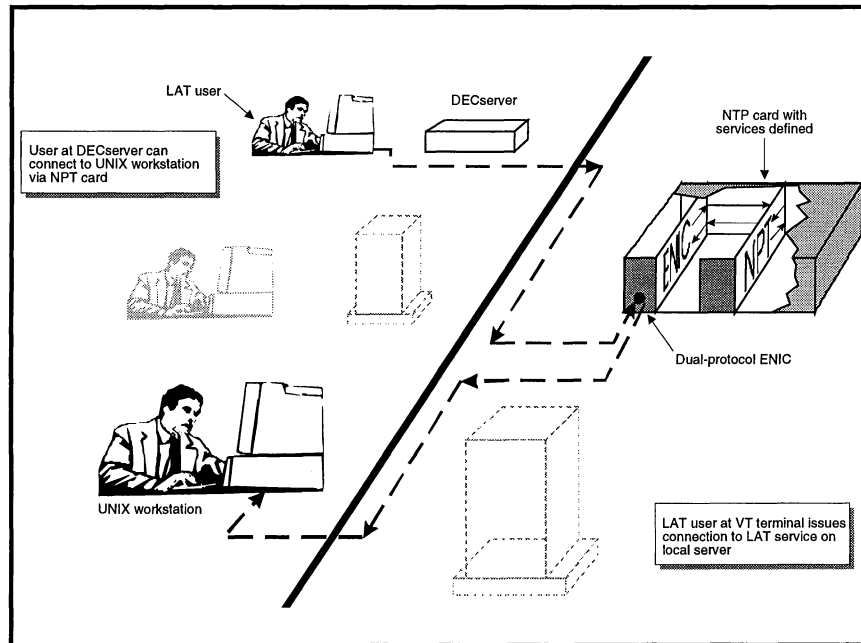


Figure 2-25. User connects to workstation via NPT card

Providing Access to a UNIX Workstation

This section explains how to configure a LAT service on a server to provide access to a UNIX workstation acting as a Telnet host (figure 2-25):

Execute:

```
Local> change service {name of lat service} port {number of
        logical server port} identification "{string of text}"
        virtual enable virtual {ip address of destination
        Telnet host} [ENTER]
```

Then execute:

```
Local> change ports {numbers of incoming and outgoing server
        ports} access dynamic virtual enable [ENTER]
```

Example: On a Communications Server with a 32-port Line Card in slot one (see table 2-1), allow one session to be translated from the LAT service *SUN1* to the TCP/IP address 123.45.6.7, execute:

```
Local> change service sun1 port 33 identification "unix on the
        sun" virtual enable virtual "123.45.6.7" [ENTER]
```

Then execute:

```
Local> change ports 33,49 access dynamic virtual enable [ENTER]
```

Configuring a LAT Service to Translate to a TCP/IP Host

Since the association to the IP address is made on the service definition, multiple services can share the same group of ports.

To configure a second LAT service (offered on the same server) to translate to a second TCP/IP Telnet host at another IP address (figure 2-26), execute the following additional command:

```
Local> change service {name of lat service} port {number of  
logical port} identification "{string of text}"  
virtual enable virtual "{ip address of second Telnet  
host}" [ENTER]
```

Example: To configure the LAT service *SNEEZY* on a Communications Server with a 32-port line card to translate to a Telnet host, (in this example, a Telnet host with IP address 123.45.6.7), add the following step to the single service configuration described above:

```
Local> change service iris port 33 identification "Telnet  
Service Sneezy" virtual enable virtual "123.45.6.7"  
[ENTER]
```

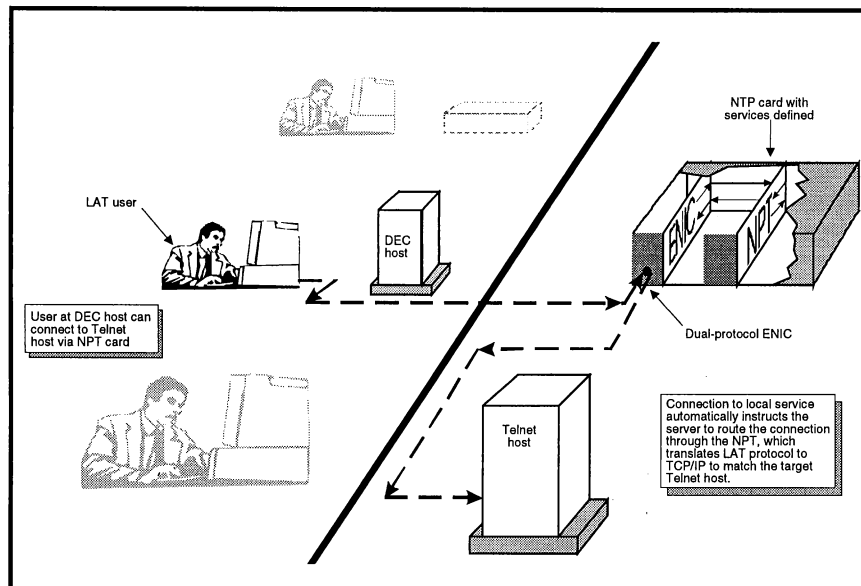


Figure 2-26. User connects to Telnet via NPT card

Connecting to LAT-Translated-to-Telnet Services

To connect to LAT-translated-to-Telnet services, execute the same command used to connect to a normal LAT service; that is:

```
Local> connect {name of service} [ENTER]
```

Just as you can set LAT services to provide access to Telnet hosts, you can also configure a Telnet IP address to provide access to a

LAT service (figure 2-27). To configure a Telnet host to translate to a LAT service, execute:

1. Local> **change address** {IP address of server as Telnet source host} [ENTER]
2. Local> **change service** {name of destination lat service on DEC host} port {number of incoming npt port on server} lat disable Telnet enable ip {ip address of server as Telnet source host} virtual enable virtual "{name of destination lat service}" tcp port 23 [ENTER]
3. Local> **change port** {number of incoming and outgoing npt ports} access dynamic virtual enable [ENTER]

where:

IP ADDRESS OF SERVER—Identifies the server as a Telnet host, which can accept incoming IP connections to a logical NPT port.

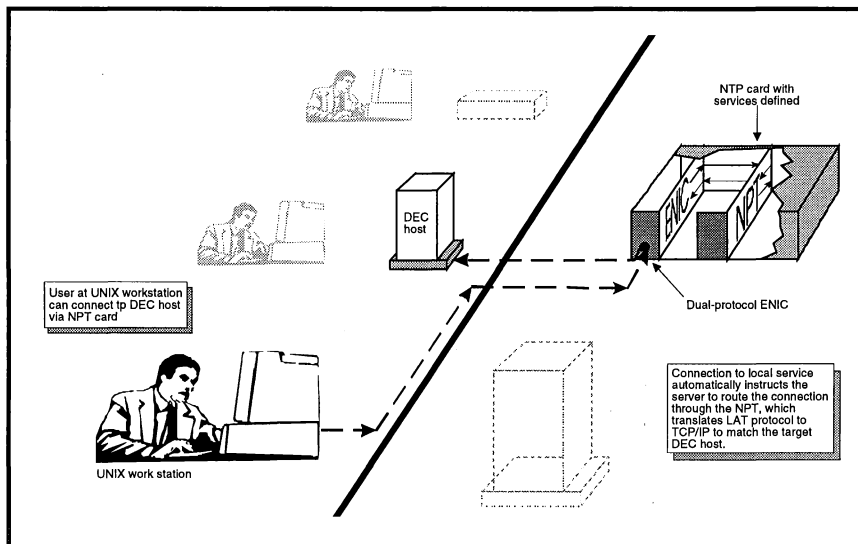


Figure 2-27. UNIX user can connect to DEC via NPT card

NAME OF DESTINATION LAT SERVICE—Identifies the local LAT service accessible through the server, which is acting as a Telnet host.

NUMBER OF INCOMING NPT PORT ON SERVER—Identifies the server NPT port, which accepts incoming Telnet connections for translation to the destination LAT service.

LAT DISABLE—Deactivates LAT protocol at the server port.

TELNET ENABLE—Activates TCP/IP protocol at the server port.

VIRTUAL ENABLE—Is set on the logical NPT port with which the local service is associated. The Virtual Enable feature automatically loops you from one port to another. That is, if you log out of the remote NPT port, you are not really logged out. Instead, you remain at the remote NPT port as long as Virtual is disabled at the remote NPT port; you are not returned to the local server NPT port. Then, when you press **ENTER** a few times, you are returned to the Telnet host with which you were connected.

Note: If you want the **logout** command to return you to the local server NPT port, you also must set the remote NPT port to Virtual Enable. Then, when you log out of the remote NPT port, you are returned to the local server NPT port.

VIRTUAL “NAME OF DESTINATION LAT SERVICE”—Specifies the LAT service name to which the incoming Telnet connection automatically is translated.

TCP PORT 23—Defines TCP standard communications. Recall that whereas LAT looks for service names, Telnet searches for IP addresses and associated TCP port numbers. By convention, when you connect to a host at an IP address, and you do not specify a destination port number, the connection *automatically* is forwarded to that host at TCP port 23. When more than one host resides at a single IP address, you must define a TCP port number for each NPT host and users must add that destination port number to the **connect telnet** command. That way, the connection can be completed to the proper host. Alternatively, you can assign a different IP address to each host and connect to the proper host without specifying a TCP port number.

Example:

Using a Communications Server with a 32-port line card, to allow an incoming Telnet session to be translated and forwarded via the IP address 123.45.6.7 to the service VAX8700, execute the following commands:

```
1.Local> change address 123.45.6.7 [ENTER]
```

```
2.Local> change service {name of service} port 33 lat disable  
Telnet enable ip 123.45.6.7 virtual enable virtual  
"vax8700" tcp port 23 [ENTER]
```

```
3.Local>change port 33,49 access dynamic virtual  
enable [ENTER]
```

PORT 33—Identifies the server NPT port, which accepts incoming Telnet connections for translation to the destination LAT service.

VAX8700—Is the LAT service to which the Telnet user's host forwards data.

TCP PORT 23—If the server only has one IP address, you must tell it which TCP port is connected to the destination Telnet host. If you do not specify a destination port number, the connection is *automatically* forwarded to the host at TCP port 23. In this example, TCP port 23 is specified.

Port 33,49—Identifies the pair of server ports through which the translation process occurs.

Connecting to Telnet-Translated-to-LAT Services

To connect to this Telnet host that will translate to a LAT service, execute:

```
Local> connect Telnet 123.45.6.7/23 [ENTER]
```

When you execute this command, the server searches its database for all locally defined services for which telnet is enabled and picks the LAT service associated with TCP port 23.

Chapter 3

Operating the Communications Server

Introduction

After you have installed the network interface and device cards (see the *Communications Server Installation Guide*), you are ready to begin operations. You can conduct operations from either the Communications Server front panel or an attached terminal. Some functions are easier to perform from the front panel using the control keys and onboard display; i.e., running diagnostics. You probably will find it easier to enter lengthy commands with a full-size keyboard and attached terminal.

This chapter contains the following topics—

- Running startup diagnostics
- Executing server commands from the front panel
- Logging into the server through the parser
- Logging out of the server
- Using server online help
- Connecting to a LAT: a Telnet service
- Fingering a user on the network
- Session control

All functions performed through the front panel can also be done through a terminal at the server's parser (`Local>` prompt) level. For example, you can initialize the server by pressing the **ENTER** key twice on the attached terminal.

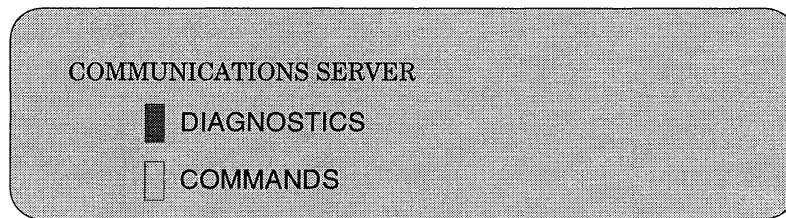
Press any of the front panel touch control keys to activate the liquid crystal display (LCD) screen. Use the LCD screen to monitor your activities and results. Once the screen is active, initialize the server. Verify the proper installation of all server components by executing diagnostic procedures before beginning operations. Correct any abnormalities before proceeding. Then, begin executing server commands either through the front panel or through an attached terminal. While the server is running, periodically monitor the network activity and status lights to confirm that operations are proceeding smoothly.

Running Startup Diagnostics

Power *on* the server by pressing the rocker switch at the rear of the unit. The server initialization and self-test period begins. You may either execute diagnostic routines or server commands from the server front panel.

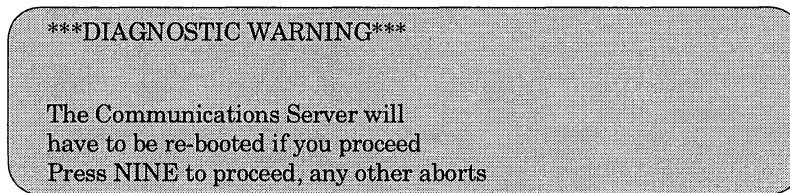
The server maintains its own software that it uses during the initialization process. Thus, the server does not need to connect first to a network load host to obtain a download of its configuration software image before booting itself. The server boots itself from its own onboard software. As an option, the server can be booted from an outside source, such as a host or a *SmartRAMCard*. See *Understanding Software Uploading Options* in chapter 4.

The initial menu screen shows the box beside the Diagnostics option darkened. See figure 3-1. Press any cursor key to switch from one option to the other.



3-1. Initial menu display

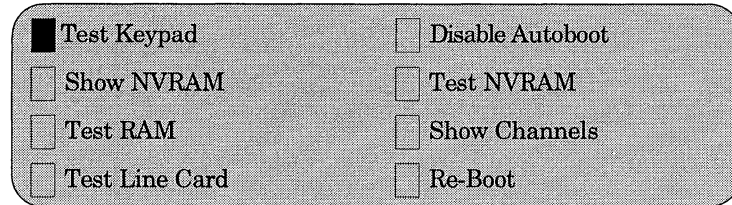
Select the Diagnostics option and press **ENTER**. The screen shown in figure 3-2 appears:



3-2. Diagnostic warning display

If you press **9 Y Z**, the main diagnostic menu screen appears. See figure 3-3.

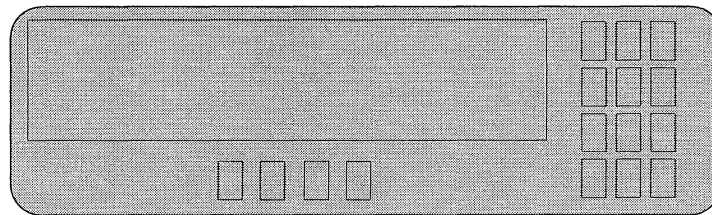
The darkened box indicates the selected option. Press the arrow keys to move the indicator to the desired box. Press **ENTER** to execute the selection.



3-3. Main diagnostic menu

Keypad Tests

Select Test Keypad and press **ENTER**. The screen shown in figure 3-4 appears.



3-4. Test keypad display

The server tests the integrity of the keypad hardware and related circuitry. To test each of the front panel keys, press the key corresponding to each control key box. Listen for a tone and wait for the key to darken. As each key test is completed, continue with the next key. A message at the end of the test will signal successful completion. When you see the message *Keypad Test Completed Successfully*, press any key to return to the main diagnostic menu.

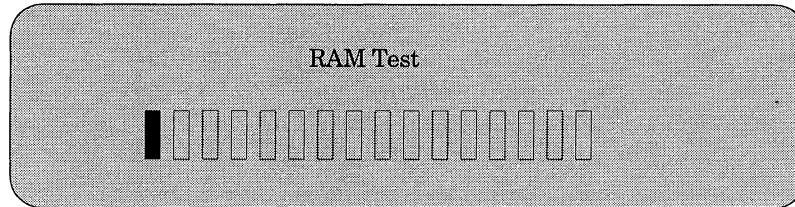
Canceling the Key Test

To cancel the keypad test; that is, to exit the test before pressing each key, and return to the main diagnostic menu, press **ENTER** three times.

NVRAM Status

Select Show NVRAM and press **ENTER**. The server automatically executes the program that tests the *NVRAM*. If the test is completed without error, the message *NVRAM Status - VALID* appears. Press any key to return to the main menu.

Press any key to return to the main diagnostics menu.

**3-5. RAM test display]**

RAM Test

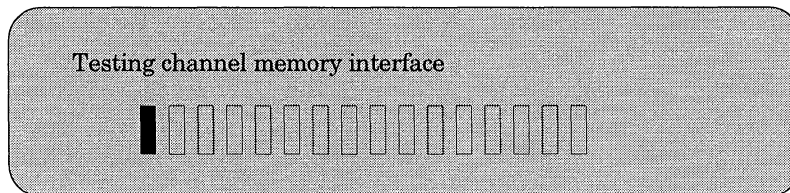
Select Test RAM and press **ENTER**. The screen in figure 3-5 appears.

To indicate that the test is in progress, the blackened box moves from left to right, across the field of 16 boxes. Then, it changes direction to complete the cycle. After completing the test, if no errors are detected, the message RAM Test Passed appears.

Press any key to return to the main diagnostics menu.

Line Card Tests

Select Test Line Card and press **ENTER**. When the message Enter the slot # of the line card is visible, select the slot number of the card that will be tested. Slots are numbered from one to four, with slot number one being closest to the ENIC. Enter that value and press **ENTER**. The server immediately begins the test program, and the screen displayed in figure 3-6 appears.

**3-6. Channel memory interface test**

To indicate that the test is in progress, the blackened box moves from left to right, across the field of 16 boxes. Then, it changes direction to complete the cycle. After completing the test, the screen shown in figure 3-7 appears. The information in this screen varies depending on the type of line card you have installed in the slot.

Memory Interface	→ PASSED
Backplane Interrupt	→ PASSED
Onboard Diagnostics	→ PASSED

3-7. Successful completion of line card tests

Press any key to return to the main diagnostics menu.

Server Autoboot Function

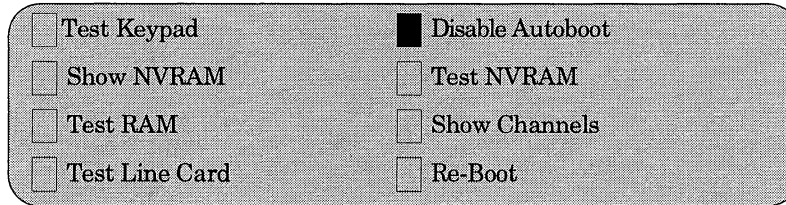
The server Autoboot function is a switch that you can use to select alternative booting sources to onboard ROM. Enable the Autoboot to activate the server primary and secondary boot options which you can use to assign user-definable primary and secondary boot sources. Execute the **Change / Define Server Primary / Secondary Boot** commands to specify the primary and secondary boot sources (see Software Uploading Options in chapter 4). The *SmartRAMCARD* is the default primary booting source. The default secondary booting source is the onboard ROM. Disabling the Autoboot function instructs the server only to use the onboard ROM as the boot source.

To disable the autoboot function, select Disable Autoboot and press **ENTER**. The server then disables the Autoboot function, and it displays Enable Autoboot in place of the previous Disabled Autoboot option. Select this option to allow for user-definable primary and secondary boot sources. When the server front panel displays the Disable Autoboot option, it means the server autoboot function is currently enabled. Select this option to instruct the server only to use the onboard ROM as the boot source. See figures 3-8 and 3-9.

Operation

<input type="checkbox"/> Test Keypad	<input checked="" type="checkbox"/> Enable Autoboot
<input type="checkbox"/> Show NVRAM	<input type="checkbox"/> Test NVRAM
<input type="checkbox"/> Test RAM	<input type="checkbox"/> Show Channels
<input type="checkbox"/> Test Line Card	<input type="checkbox"/> Re-Boot

3-8. Main menu with Autoboot enabled

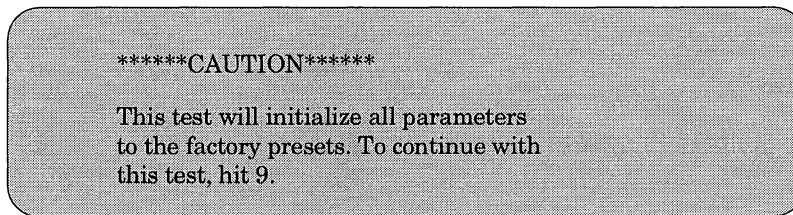


3-9. Main menu with Autoboot disabled

NVRAM Tests

Select Test NVRAM and press **ENTER**. The server is instructed to test its non-volatile (or permanent) data base, and the screen shown in figure 3-10 appears. Press any other key to abort the NVRAM test.

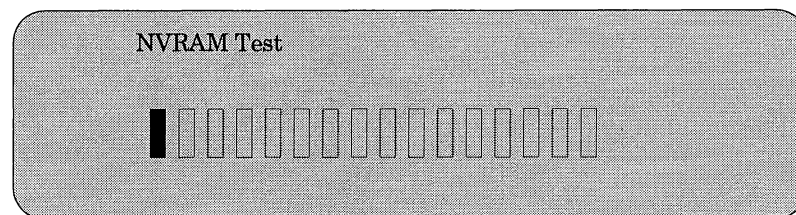
The cautionary note is displayed to inform users of the consequences of the NVRAM test. The test re-initializes all server, port and service characteristics to



3-10. NVRAM Test Caution

their factory settings. This erases any changes that may have been made to the factory presets.

To execute the *NVRAM* test, press the **9YZ.** key. The screen in figure 3-11 appears.



3-11. NVRAM test display

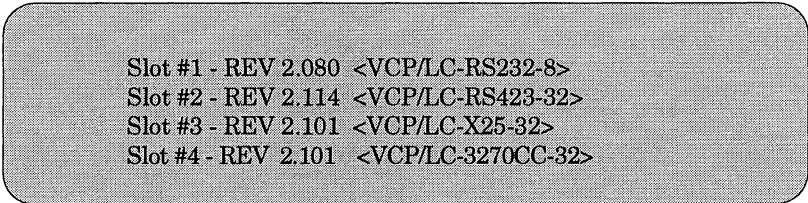
The test continues without additional user assistance. When the test is complete, and no errors are encountered, the message **NVRAM PASSED** appears on the display.

If an error is indicated, a possible cause can be a discharged NVRAM battery. For further information, see the *Communications Server Hardware Installation and Service Manual*.

Press any key to return to the main diagnostic menu.

Line Card Slot Tests

Select **Show Channels** and press **ENTER**. The server checks each of the circuit board slots to determine which circuit boards are installed in the chassis. The screen in figure 3-12 appears.



```
Slot #1 - REV 2.080 <VCP/LC-RS232-8>
Slot #2 - REV 2.114 <VCP/LC-RS423-32>
Slot #3 - REV 2.101 <VCP/LC-X25-32>
Slot #4 - REV 2.101 <VCP/LC-3270CC-32>
```

3-12. Installed line card and slot location display

The screen displays different information depending upon the line cards that are installed in the Communications Server.

Press any key to return to the main diagnostic menu.

Server Reboot

To restart the server, select **Re-Boot** and press **ENTER**. If the restart procedure is completed without error, the server redisplay the initial menu.

Executing Commands from Front Panel

This section explains how to enter commands into the server via the front panel keypad. To log into the server using a terminal and to begin executing commands through the server parser, turn to *Logging Into the Server Through the Parser*, later in this chapter.

When the server is booted and the initial menu screen (figure 3-1) is displayed, the box next to the Diagnostics option is highlighted. Press any cursor key to switch to the Commands option. Press **ENTER** to display the command menu.

About the Command Menus

The server command structure is hierarchical; that is, a complete command consists of a string of separate parts, each taken from a subsequent level in the hierarchy. The highest level, level one, is a root operator, which appears first in the command string. Level two is the object of the command, which is directed by the root operator. Level three is a characteristic, which modifies the object. Level four is a variable, which modifies the characteristic. For example, consider the following command:

Server> **SET PORT 6 ACCESS REMOTE**

where:

SET—Is the root operator

PORT 6—Is the object

ACCESS—Is the characteristic

REMOTE—Is a variable

The advantage of the hierarchical structure is its simplicity. From just a few basic roots, one can create many different combinations of objects and modifiers to devise all of the hundreds of possible commands that are executable through the server (via either the front panel or an attached terminal).

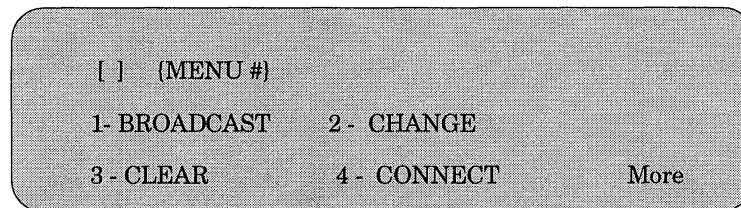
Note: All commands executable from the front panel also can be entered at the server Local> prompt via an attached terminal. chapter 2, *Configuring the Server*, and chapter 4, *Managing the Server*, explain how to enter these and other commands at the Local> prompt.

The first level contains a menu of twenty-one root operators. Similarly, each subsequent level of the hierarchy contains a new menu of available choices. Use the control keys to navigate through the menus and to build command strings to execute the commands. The functions of the control keys are shown in table 3-1. For example, press the right arrow [→] to display the next portion of the menu. Press the up arrow [↑] at any time to return either to the initial menu or to the previous hierarchical level.

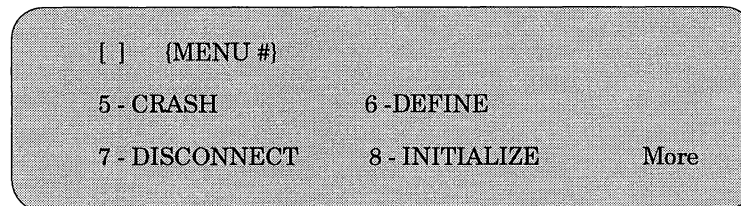
After you select a root operator from the first hierarchical level, the server automatically displays the next level in the hierarchy; that is, the objects associated with that root. The server automatically displays the subsequent level of the hierarchy following each selection.

LCD Menu Displays

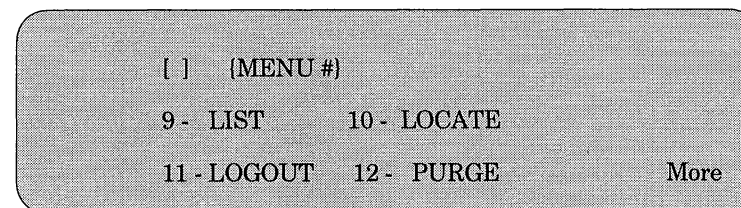
The server front panel keypad permits entry of many server commands. Figures 3-13 through 3-18 illustrate each of the displays in level 1 of the hierarchy. Not all parser commands are executable from the front panel. Use a keyboard attached to the ENIC to execute any command not found on the front panel.



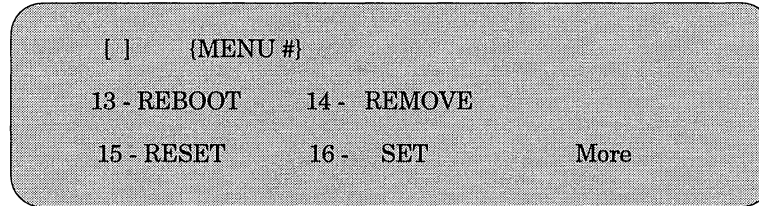
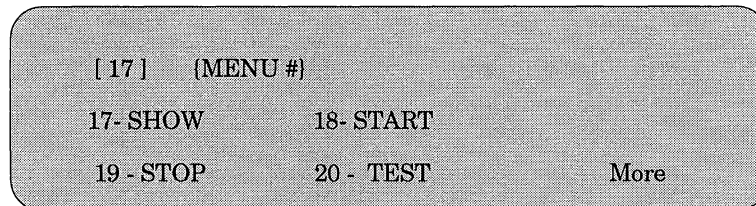
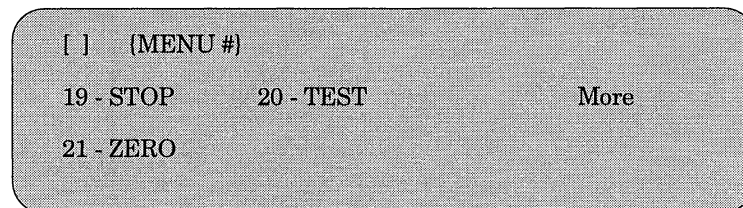
3-15. Level 1, display 1



3-13. Level 1, display 2



3-14. Level 1, display 3

**3-16. Level 1, display 4****3-17. Level 1, display 5****3-18. Level 1, display 6**

Correcting Improperly Entered Commands

If, at any time, you execute a command improperly, the server displays the screen shown in figure 3-19. Recheck your entry and re-enter it correctly.

Then, re-execute the command.

```

Server>
Local -701- Command Syntax Error
Server>

```

3-19. Error message display

Entering Show Events Command from Panel

The server Events Log retains a description of each transaction that the server was unable to complete. Thus, it is a helpful source of information during troubleshooting activities. Following is a procedure for executing the **Show Events** command from the front panel.

1. Power on the Communications Server. The initial menu screen is displayed.
2. Press any cursor key to move from the diagnostics option to the commands option.
3. Press **ENTER**. The first root operator menu screen is displayed. See figure 3-13.
4. Press the right arrow [→] key four times to see the **Show** command. See figure 3-17.
5. The **Show** command is number 17. Press the alphanumeric key labeled **1ABC** once. The numeral one appears between the brackets in the upper left-hand corner of the screen display. Press the alphanumeric key labeled **7STU** once. 17 now appears between the brackets in the upper left-hand corner of the screen display. See figure 3-20.

To summarize: To select any command, note the number beside the command. Press the key (or keys) that correspond to the number. If you execute an incorrect

```

[17]  (MENU #)

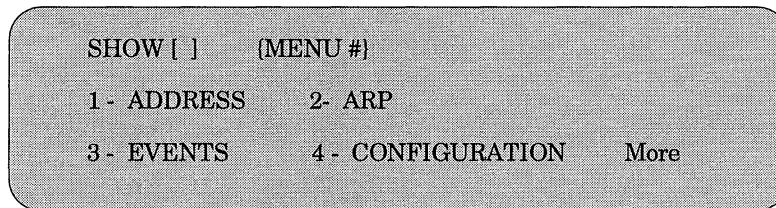
17- SHOW      18 - START
19 - STOP     20 - TEST      More

```

3-20. Selecting 17-SHOW display

numeral, press the delete key once to remove the numeral from the brackets that enclose the command number.

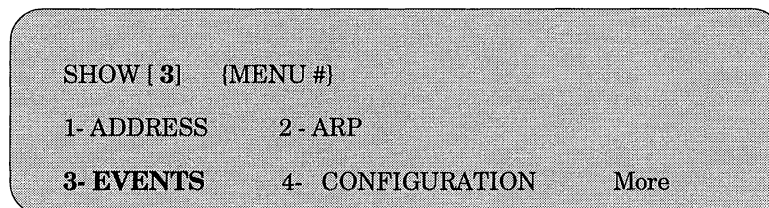
6. Press the down arrow [↓] key to send the entry to the server for validation. If the entry is valid, the server displays the root operator display (figure 3-21) and prompts the user for an object.



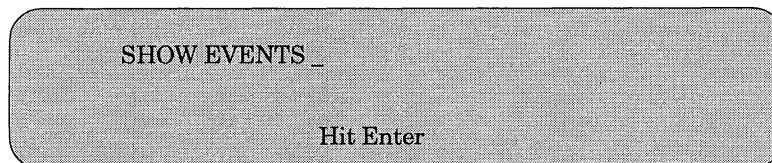
3-21. Object level of Show menu

If you wish to move from the current level in the hierarchy to the previous level, press the up arrow [↑] key.

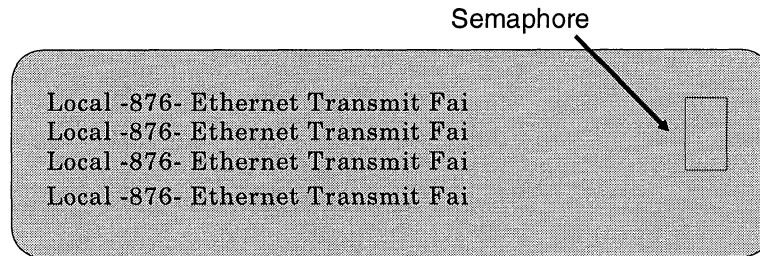
7. Press the alphanumeric key labeled **3GHI**. The numeral three appears between the brackets in the upper left-hand corner of the screen display.



3-22. Events selected from menu



3-23. Command before execution

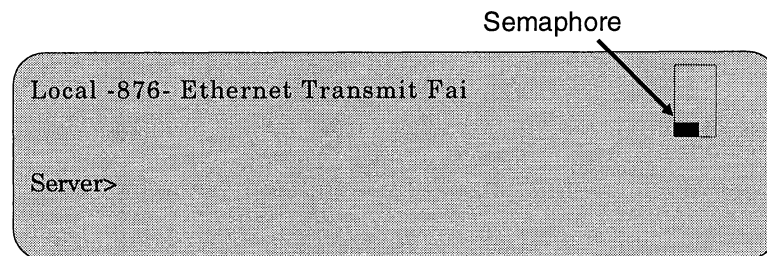


3-24. First subset of of Events log on LCD screen

8. Press the down arrow [↓] key to send each entry to the server for validation, and display the next prompt.
9. To execute the command press **ENTER**. A list of the Events Log messages is displayed.

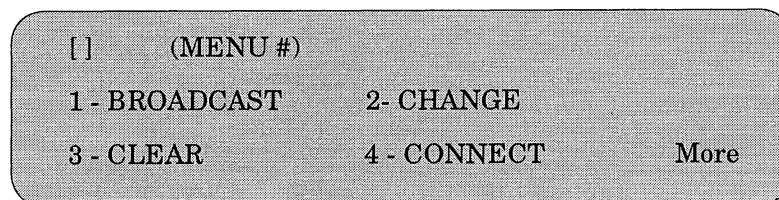
The LCD screen shows only the twenty-one most recent messages. To see events prior to these messages, log into a printing terminal and print the Events Log. All messages generated since the last re-initialization are printed.

10. Use the four navigation keys to view all of the available messages. At the end of the list, the **Server>** prompt is displayed as shown in figure 3-25.



3-25. Events subset of Events log display

11. To exit the Events Log display, press **ENTER**. The server again displays the root operator menu. See figure 3-26.



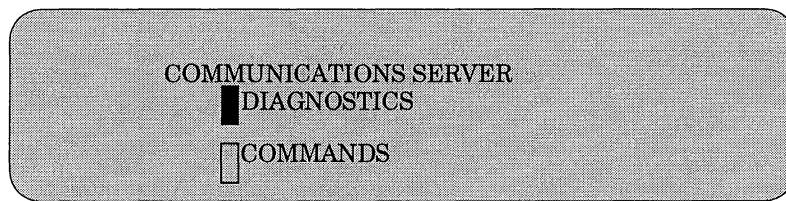
3-26. Level 1, display 1

Viewing Broadcasts

A user operating through another local port may send a broadcast message to the server. If either the initial menu screen or the root operator menu is currently displayed, the server receives the broadcast message. However, if the server is currently displaying any of the diagnostics screens, it does not receive the broadcast message.

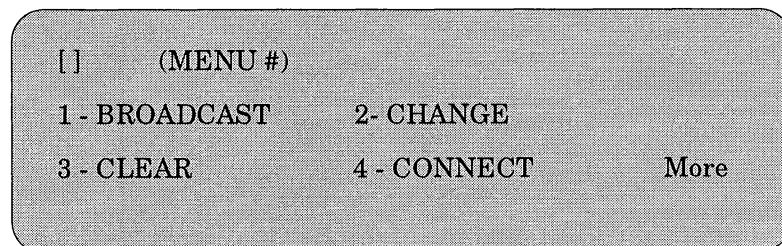
If the LCD screen is dark, upon receiving the broadcast message, the LCD backlighting is activated, and the server simultaneously emits an audible beep. The image remains on the LCD screen even if the LCD backlighting is not active. If the server is currently displaying the initial menu, and you wish to view the message, execute the following procedure from step one. If you are viewing any of the command menu options, execute the following procedure from step three.

1. At the initial menu screen, press any cursor key to highlight the box beside the Commands option. See figure 3-27.

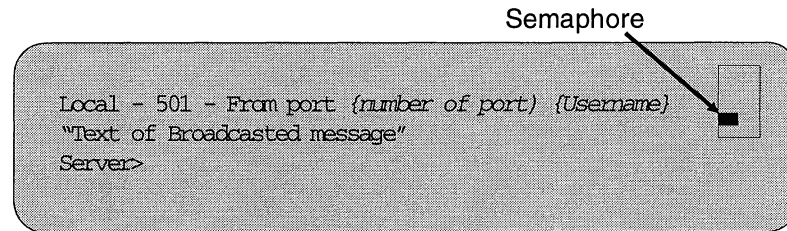


3-27. Initial menu display

2. Press **ENTER**. The first root operator menu screen is displayed as shown in figure 3-28.



3-28. Level 1, display 1



3-29. Broadcast message display

3. Press **ENTER** again. The server displays the region of the limited-view display containing the broadcasted message. Note the position of the semaphore in figure 3-29.
4. To return to the initial menu screen, press the **ENTER** key again.

Logging into Server through Parser

This section explains how to use a terminal attached to the server to log into and use the server. The LCD display and related keypad on the front panel cannot be used to enter all server commands. And many commands can be entered much more quickly and easily with a keyboard and terminal. If you do not already have a terminal attached to the server, refer to the *Communications Server Installation and Service Guide* for instructions.

Logging into the Server

You must log into a server port to use server facilities. It is assumed at this point that the server is powered up and initialized. Proceed as follows:

1. If you cannot see the `Enter Username>` prompt, awaken your server port by pressing:

[ENTER] [ENTER]

Note: If there is no response, ensure that the terminal is connected to a port on the server and that the port is properly configured for the terminal. If necessary, try another physical port.

2. At the `Enter Username>` prompt, type your port user name.

Please type **HELP** if you need assistance.

`Enter Username> {port user name} [ENTER]`

Note: If your port does not have an assigned user name, type your own name (up to 16 characters) at the prompt.

3. Wait for the `Local>` screen prompt.

Logging out of the Server

When finished, to disconnect all sessions, log out of the server. Proceed as follows:

1. Break out of your current application and return to the `Local>` prompt.
2. Execute the **logout** command:

```
Local> logout [ENTER]
```

3. Wait for a message verifying completion of your logout process.

Using Server On-line Help

When you are logged into the server, you also have two on-line options:

Add a question mark (?) to the end of the command for a list of options available.

Access the server on-line help feature from the `Local>` prompt.

To access on-line help, proceed as follows:

1. Return to the `Local>` prompt. Execute the **Help** command.

```
Local> help [ENTER]
```

2. Select (from the displayed listing) the specific topic you would like information. For example, for information about the **Forward** command, type:

```
Local> help forward [ENTER]
```

3. Exit on-line help, and return to the server prompt by typing:

```
Help> [CTRL] [Z]
```

Connecting to a LAT Service

To connect your terminal to a LAT service, proceed as follows:

1. Access the `Local>` prompt.

```
Local>
```

2. Display available services. Execute:

```
Local> show services [ENTER]
```

3. Examine the list of available LAT services. Choose the service with which you wish to connect. Execute:

```
Local> connect {name of LAT service} [ENTER]
```

4. Wait for a message requesting your user login and, if required, a user password. If the server prompts for a password, execute:

```
Password> {service password} [ENTER]
```

5. If the password is valid and is correctly entered, the server will execute the connection attempt. If the connection is completed, the destination service prompt is displayed.

Connecting to a TELNET Host

To connect to a Telnet host, your local server *must* have an IP address. To determine the IP address of the local server, execute the **Show Address** command. If the local server has no IP address, assign it one. See *Configuring the Server*, chapter 2.

1. Access the Local> prompt.

```
Local>
```

2. Display available services. Execute:

```
Local> show services [ENTER]
```

3. Examine the list of available Telnet services. Choose the host service with which you wish to connect. Execute:

```
Local> connect {name of TELNET host} [ENTER]
```

4. Continue from step four in *Connecting to a LAT Service*, above.

Fingering a User on the Network

Execute the Finger command to display information about users currently logged into local or remote hosts.

Execute:

```
Local> finger {username}@{hostname} [ENTER]
```

or

```
Local> finger {username}@{ip address} [ENTER]
```

Where:

USERNAME—Is the login name of the user.

HOSTNAME—Identifies the remote host at which the target user is logged in.

IP ADDRESS—Is the address of the remote host at which the target user is logged in.

Session Control

The server supports several sessions operating at the same time. Multiple sessions can be started and maintained either to one service or to several services. Following are procedures to control session operation.

Connecting to an Additional Session

1. To start an additional session to a LAT service or Telnet host, return to the Local> prompt, and execute one of the following commands:

```
Local> connect {name of LAT service} [ENTER]
```

or

```
Local> connect {IP address of telnet host} [ENTER]
```

or

```
Local> connect {domain name of telnet host} [ENTER]
```

Note: You need not add the keyword “Telnet” to the connect command (e.g., connect telnet doc). The server automatically connects via Telnet if it cannot complete the connection via LAT.

2. Log into the service or host.

Determining the Number of a Session

Sessions are identified numerically. To display the number that identifies the session in which you currently are operating, return to the Local> prompt and execute the following command:

```
Local> show session [ENTER]
```

Changing Between Sessions

When you are connected to more than one session and you want to change from your current session to another, use the **Forwards** and **Backwards** commands to execute the change.

The **Forwards** command moves you to the next higher-numbered session. For example, if you are operating three sessions and you currently are operating in session *two*, the **Forwards** command moves you to session *three*. Executing the **Forwards** command in session *three* moves you to session *one*.

The **Backwards** command moves you to the next *lower*-numbered session.

If you are only connected to two sessions, each execution of either the **Backwards** or **Forwards** command moves you to the alternate session.

Another way to change your current session is to execute:

```
Local> resume session {number of session} [ENTER]
```

Returning to Local Mode from a Session

When you are connected to a LAT service or a Telnet host, your service port is operating in *service mode*. That is, your terminal operates as though it were directly connected to the target host. When you are *not* connected to a target host, but are logged into the local server port, your port is operating in *local mode*.

To return from service mode operation to local mode operation, proceed as follows:

1. If you are not using a modem, press **BREAK**. If you are using a modem, go to step *two*.
2. Determine if the local switch character has been defined. **BREAK** is the default key for returning to local mode. Some modems use **BREAK** to end a connection. If you are using a modem, *do not press BREAK* to return to local mode.
3. Press the local switch character to return to local mode.

Note: When you return to local mode, the session you just left is suspended and considered inactive. However, that session remains your current session because it is your most recently-active session.

Resuming a Session from Local Mode

To return to the session you just left, which is your most recently-active session, return to the Local> prompt and type:

```
Local> resume [ENTER]
```

Disconnecting from the Service Mode

If you currently are operating in service mode and you want to end your current session, execute the application command that will end your current process. For example, if you are logged into a session on VAX/VMS, execute the Logout command. See your host application's user's guide for more information.

Disconnecting from the Local Mode

If you currently are operating in the local mode and you want to end your current session, return to the Local> prompt and type:

```
Local> disconnect session {session number} [ENTER]
```

Note: You cannot return to a session using the **Resume** command after you have used the **Disconnect** command.

Chapter 4

Managing the Communications Server

This chapter contains the following topics—

- Autoboot selection
- Auto-Ethernet selection
- Centralized Accounting and Reporting System (CARS)
- File transfer between PCs and hosts
- Managing the IP address pool
- Macro facility
- Storing server configuration parameters
- Software uploading options
- Time and date management
- Tracing a route on the network
- Upgrading server software via software keys
- UNIX command set

All server data base entries require privileged user status. To set your user port to privileged status when you log into the server, execute:

```
Local> set privileged [ENTER]  
Password>
```

At the Password> prompt, enter the privileged password. The server factory default password is *system*. You can change this password to a new string if you want to do that.

Managing the Server

Autoboot Selection

The Server Autoboot function is a switch that you can use to select alternative booting sources to onboard ROM. Configure the autoboot feature as follows:

```
Local> change server autoboot {enabled} {disabled}  
      {warm} [ENTER]
```

Choosing *enabled* (the default) activates the server Autoboot function and allows you to specify primary and secondary sources from which the server can obtain configuration images. *Disabled* deactivates the function. *Warm* is a diagnostics feature to restart the server without rebooting; that is, when the command is executed, the server restarts itself using all of the current configuration values. For more information, refer to *Software Uploading Options* in this chapter.

Auto-Ethernet Selection

Server Ethernet connector selection is automatic and can be controlled with the following command:

```
Local> change server ethernet {auto-select}  
      {thinwire} {aui} [ENTER]
```

Thinwire and *aui* are variables, each representing a type of Ethernet connector on the server. Autoselect senses which of the possible Ethernet adapters is active, and automatically sets the server for that interface. If you do not specify otherwise, the server factory default setting is Auto-Select.

Centralized Accounting and Reporting System

The Centralized Accounting and Reporting System (CARS) monitors server activity and keeps a tally of the following entries:

- All current eventlog messages
- All successful and unsuccessful attempts to log in and out of the server
- All sessions established including the protocol used in that session
- Any session failures
- Entries not maintained because of memory limitation

These entries, along with the date and time at which they occurred, are kept in the CARS data base and can either be sent to a specified port, or saved to a designated file located on a *SmartRAMCARD*.

Viewing CARS Parameters

Use the following command to view the CARS parameters. Execute:

```
Local> show cars [ENTER]
```

The CARS parameters are explained briefly in the following section.

Altering CARS Parameters

The following is an explanation of how to set up all available CARS parameters. Not all of the following parameters have to be altered as there are defaults for many parameters.

1. Enable or disable the Autoclear function. Autoclear determines whether the local server automatically clears the CARS entries. Execute:

```
Local> change cars autoclear {enable} {disable}  
[ENTER]
```

2. Determine a destination for the CARS data output. The destination can be—

- PORT (send output to specified local server port)
- FILE (send output to a specified local server file)
- TCP (send output to a TCP socket)
- UDP (send output to a UDP socket)

Execute:

```
Local> change cars destination {port} {file} {tcp}  
{udp} [ENTER]
```

3. Determine a format for the CARS output file. The output file can be—

- HUMAN (ASCII characters)
- RECORD (column record format)
- SYSLOGD (the format for sending to TCP or UDP socket)
- DIF (VisiCalc™ data)

Execute:

```
Local> change cars format {human} {record} {syslogd}  
{dif} [ENTER]
```

4. Either enable or disable account collection and transmission. Essentially, this either activates or deactivates CARS. By default CARS reporting is disabled. Execute:

```
Local> change cars reporting {enable} {disable}  
[ENTER]
```

5. Determine the number of events that the CARS system will accumulate before sending output to the destination. The range is from 1 to 256; where 256 is the factory default. Execute:

```
Local> change cars update counter {counter value}  
[ENTER]
```

6. Decide how long the CARS system will wait before sending output to the destination specified with the destination command (step 2). This parameter is essentially a backup to the counter parameter. For example, if you wanted a report every hour no matter how many events were received, you'd set the timer for 60. In this way, even if the specified number of events was not reached in an hour, the report still would be sent. The range is 1 to 1440 minutes. The update timer factory default is 240 minutes.

```
Local> change cars update timer {timer value} [ENTER]
```

7. Decide whether to enable wrapping and overwrite the oldest CARS output in memory, or to disable wrapping and stop recording new CARS output when the memory is filled. Execute:

```
Local> change cars wrap {enable or disable} [ENTER]
```

File Transfer Between PCs and Hosts

A personal computer that is connected to a local access port on the server can operate either in terminal emulation mode, which permits users to log into services on the Ethernet network, or in file transfer mode, which permits a personal computer to transfer files to a host. Note that in file transfer mode, both the PC and the host must support the same file transfer application.

The file transfer application (such as RAF™) allows a user of a personal computer that is attached to a server port to send files over the LAN. The personal computer has the ability to control data transparency (i.e., allowing the software in the PC to intercept special characters, such as flow control).

The server manager establishes the personal computer, which is the initiator of the session, for connection to a partner. The partner can be either a host node, a non-LAN host or a second personal computer set up as a service. The destination (or receiving)

host processor may require additional software and hardware adjustments before it is available to support the file transfer operation. Before setting up a file transfer, it is important that the server manager:

- Ascertain that the destination service in the host processor (which is serving as the partner) is available
- Verify that the file transfer program used by the partner is running

Consult your PC documentation, the service node documentation and the documentation for the file transfer program before conducting file transfers.

Configuring the Server Port for File Transfers

Following are procedures to set up a server port to transfer files to a remote host using a file transfer application.

1. To permit local users to interrupt an active connection to a local device, and thus interrupt a transfer operation process at the local device, execute:

```
Local> set port {number of target port} interrupt  
enable [ENTER]
```

When **Port Interrupt** is enabled, the break command must be set to *Local*. This way, a user whose terminal is operating in terminal emulation mode presses the **BREAK** key to interrupt a file transfer operation. At the Local> prompt, type:

```
Local> set port break local [ENTER]
```

This establishes the **BREAK** key as a local switch.

2. Connect to the destination service at the partner host processor.

```
Local> connect {name of service} [ENTER]
```

3. When the connection is completed, the file transfer application software sets the data transparency parameters, including:

- Port flow control
- Port forward switch
- Port backward switch
- Port local switch

If these switching parameters are not automatically set to the same values at both ends of the connection, reset them from the local server. Exit from the remote service and return to the Local> server prompt. Execute:

```
Local> set session passall [ENTER]
```

or

```
Local> set session pasthru [ENTER]
```

where:

SET SESSION—Deactivates **Backward** and **Forward Switch** characters for the duration of the session.

PASSALL—Instructs the server to ignore switch and flow control characters. The server will pass these characters between the personal computer and the partner as data. Note that the server will not send broadcast messages to the local port while this session is active.

PASTHRU—Instructs the server to ignore the switch characters but to recognize the flow control characters. The server will pass these characters between the personal computer and the partner as data. Note that the server will not send broadcast messages to the local port while this session is active.

4. Resume the session with the remote service. Execute:

```
Local> resume [ENTER]
```

5. Execute the file transfer operation.

If you are unable to successfully execute file transfers through the local server to a remote device (such as a host processor or a PC), recheck the local port flow control parameter settings. Consider disabling the flow control parameters as follows:

```
Local> change port flow control disable [ENTER]
```

```
Local> change port input flow control disable [ENTER]
```

```
Local> change port output flow control disable [ENTER]
```

Remember to return each of these flow control parameters to its enabled setting when you finish the file transfer operation.

6. Upon completing the file transfer, press the **BREAK** key to return to the Local> prompt.

7. Reset the transparency mode of the session and reactivate all of the special control characters, such as **Backward** and **Forward Switch**.
Execute:

```
Local> set session interactive [ENTER]
```

8. To determine if port errors occurred during the transfer, at the Local> prompt, execute:

```
Local> show port counters [ENTER]
```

Setting Circuit Timer for Transfer Rate

The Server Circuit Timer designates the time period elapsing between successive data packet transmissions from the server to the connected service nodes. While the circuit timer is running, the server collects and stores the keystrokes from each of the local ports. When the timer expires, the server transmits each packet to its respective destination node.

The circuit timer setting affects the speed at which data transfers are completed. Longer time values increase the waiting period between successive transmissions, while shorter time values decrease the waiting period.

Overall, it would appear that longer waiting periods lengthen the time required to complete the data file transmission. While users may find this objectionable, consider the alternative. Shorter time periods improve the observed terminal responsiveness, reducing the total amount of time required to transfer data, while consuming more host and CPU resources. On a lightly loaded network, excessive resource consumption may not become a problem; however, when the network traffic increases to a moderate or heavy load, such overconsumption can “bring the system to its knees,” slowing all transactions, if not stopping them altogether.

The relationship between the setting of the circuit timer, the interactive terminal response time experienced by users, and network overhead (which is directly proportional to the demand on individual service nodes) is summarized in table 4-1 below.

The optimum circuit timer value provides acceptably quick response time without overloading the network. The circuit timer default value of 80 milliseconds, which offers acceptable terminal response time without excessive network overhead, is a good compromise for moderately-loaded networks.

Table 4-1. Circuit Timer Value Relationships

Circuit timer value	Response time	Network overhead
long (200 msec)	slow	light
short (10 msec)	fast	heavy

Managing the IP Address Pool

The IP Pool feature applies to PPP links in which the remote host (such as a PC) does not have an IP address of its own. During options negotiations, before the PPP link comes up, the local server asks the remote host for its IP address. The IP address is an essential item in every packet transferred between the two devices. If the remote host responds that it has no IP address, the local server can assign one to the host. The assigned IP address is taken from a local data base of IP addresses called the IP address pool, which is maintained within the local server. The IP address remains assigned to the remote host for the duration of the link. Network users can connect to the temporary IP address of the remote host via the PPP link.

The IP address pool is populated and maintained by the local server manager. It is a suggested practice to assign at least one IP address to the IP address pool for each configured PPP link. Following are procedures to manage the IP address pool.

Adding Entries to the IP Address Pool

To add IP addresses to the IP address pool, execute:

```
Local> change ip pool {IP address to be added} [ENTER]
```

When you execute a **Show IP Pool** command, the new entry will be visible in the table.

Viewing the IP Address Pool

To view entries in the IP address pool, execute:

```
Local> show ip pool [ENTER]
```

Removing Entries from the IP Address Pool

IP addresses can be removed one at a time, or all at once, from the local server data base. To remove a single entry from the local server IP address pool, execute:

```
Local> clear ip pool local {IP address to be  
deleted} [ENTER]
```

To remove all IP entries from the local server IP address pool, execute:

```
Local> clear ip pool local [ENTER]
```

Server Macro Facility

Using the server macro facility, you can create customized programs, called macro scripts, which can be used in a variety of ways to manage the server. For example, using the macro facility, you can create a menu-driven user interface through which a user might connect to network hosts or services. An advantage of such an interface is the extra measure of security it provides—users are restricted only to the listed hosts or services. See Example 1 (following the Example of Macros table). Similarly, you can create a macro program to reconfigure selected server parameters at the user's port. When the user logs out of the port, the revised port parameters can revert to their original settings. See Example 2 (following the Example of Macros table).

A macro script is a mini-program containing both "C" programming commands and server parser commands. You can write these macro programs on a host processor or on a PC using an ASCII line editor (such as Epsilon™). The ready-to-run macro file is stored either on a *SmartRAMCARD* or on a host processor. The mini-program runs and, when it is done running, it is purged from the server. Note that in the case of the menu-driven interface script, the macro continues running in the background after it is loaded into the server. This is necessary to support the screen display until a selection is made. However, when a macro is used to configure server parameters, it terminates after writing its contents into the server memory.

Macros can be set to run either automatically or manually. In the automatic mode, as soon as the user logs into the target port, that port loads and executes a specified macro file. All of this occurs transparently to the user. In the manual mode, the port user must execute commands to read and start a specified macro.

Writing a Macro

Server macros are written in a simplified version of "C" syntax that is interpreted (as opposed to compiled). Programs are created outside the server and may be written on either a host or a PC. The resulting ASCII file is read into the server via either TFTP from a host, or, if your PC has a *SmartRAMCARD* reader, you can write the file to a *SmartRAMCARD* and then download the file from that card to the target server. Following are some general rules for writing macros:

- Control statements are *not* case sensitive; that is, you can enter both uppercase and lowercase characters.
- Put a semicolon (;) at the end of every control statement.
- You can use either single or multiple statements. A multiple statement must be enclosed with braces ({ }) to show where it ends.
- There are two types of variables: integer variables, which are signed integers, and character variables, which are strings of up to 255 characters.
- The last line of the macro file must be two dots (. .).

See table 4-2 for control statements that work with the server macro facility.

Table 4-2. Examples of Macros

Control statement	Comments
@ {server command}	Passes command to server. From within a macro, you can execute all parser commands <i>except</i> execute macro.
break	Takes the program out of "while" loop.
cls	Clears screen before you write to it.
continue	Leaves innermost loop.
else	Optional statement. If <i>ELSE</i> statement is included and <i>IF</i> statement is false, execute the <i>ELSE</i> operation and the statements immediately following it.
end	Stops macro. If macro is in a subroutine, the macro will end.
goto	Takes the program to a label somewhere within the program.
if {expression}	IF statement tests expression. If the expression is true, the program executes the statements immediately following the <i>IF</i> statement.
input {integer variable}	Use to enter variables; instructs program to locate cursor on display at coordinates x,y where x is the column and y is the row.
print {character variable}	Use to enter variables and character strings; instructs program to locate cursor on display at coordinates x,y where x= column and y= row.
return	A subroutine call statement sends program to a specified subroutine. A <i>RETURN</i> statement immediately takes program out of subroutine and returns it to the point from which it was called.
sleep {integer variable}	Pauses program for specified seconds.
while {expression}	Use this statement to create loops. You can only use comparative statements (is <i>A > B</i> ?; is <i>C = D</i> ?, etc.). You cannot use <i>OR</i> and <i>AND</i> .

Following are two examples of macro programs that you can write. You can create programs similar to these, or you can copy these programs and adapt them to your needs.

Example 1 (shown on the next page), shows a menu-driven program. Upon clearing the user's screen, the server displays a list of accessible target hosts and a prompt for the user's selection. If the user selects one of the target hosts, the program executes a connection to that host. If the user selects a choice that is out of range, the program prints an error message and repaints the screen. Note that in the case of the menu-driven interface script, the macro continues running in the background after it is loaded into the the server. This is necessary to support the screen display until a selection is made.

Example 2, shown below, is a configuration program that executes parser commands when the macro is executed. When you use a macro to configure server parameters, it terminates after writing its contents into the server memory.

```
msample()
{
    @set port 1 autobaud disable;
    @set port 2 autobaud enable;
    @set port 3 autobaud enable;
}
..
```

Disable autobaud parameter at port one and enable autobaud parameter at ports two and three

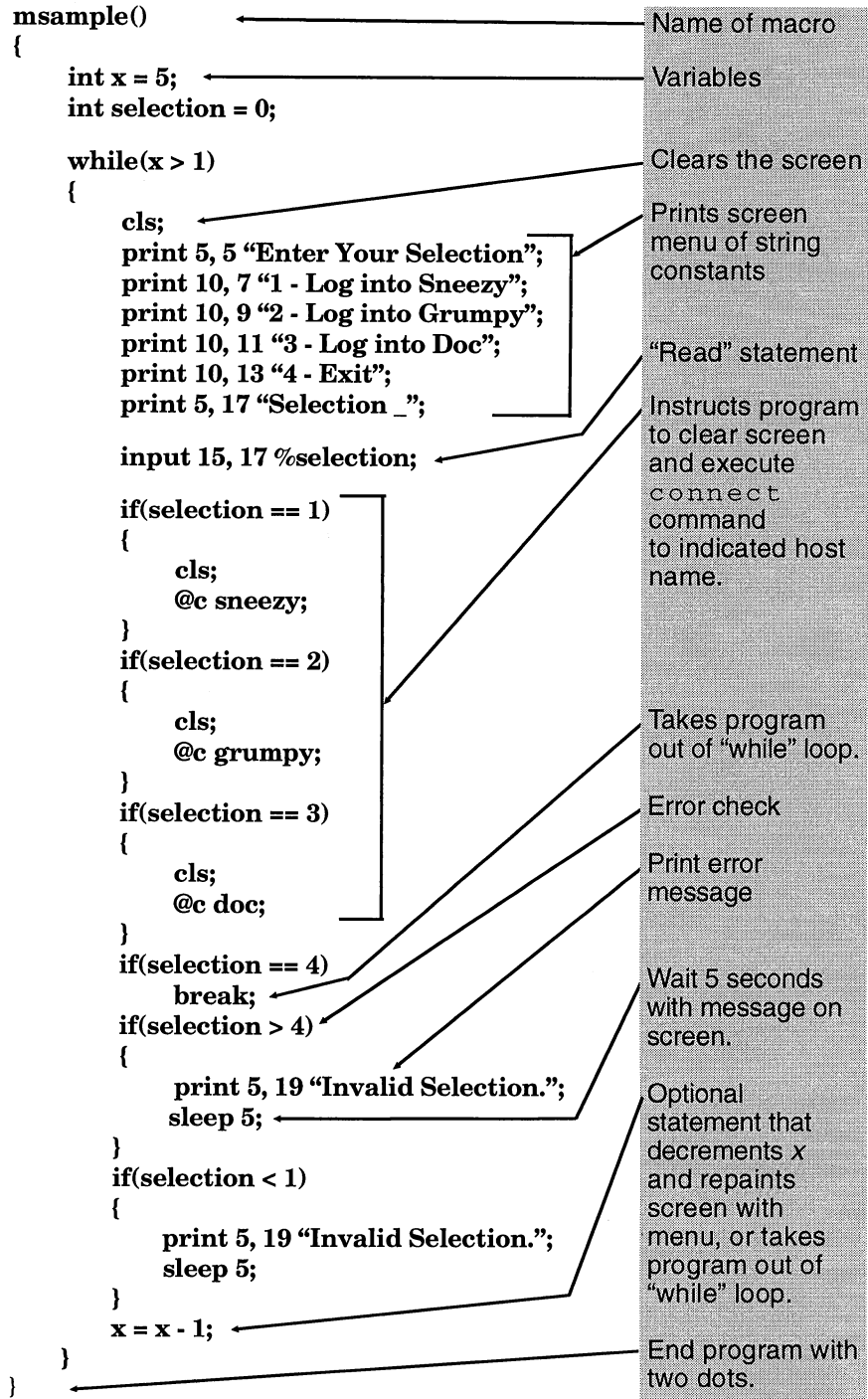
Automatically Activating Macros

Macros can be set to run automatically when the user logs into a macro-enabled server port. The user's port immediately loads and executes a specified macro file. All of this occurs transparently to the user.

To configure one or more server ports to automatically activate a macro, execute the following procedure:

1. Determine which local server ports will be configured to automatically activate a macro. These are the *target* ports referred to in the procedure.
2. You cannot activate the macro facility at a dedicated port; that is, a port that is configured to automatically connect to a service. Determine if any of the target ports are configured as such. Execute:

```
Local> show port {port no.} characteristics [ENTER]
```



3. To deactivate the dedicated and autoconnect parameters at each of the target ports, execute:

```
Local> define port {target port no.} dedicated  
       disable autoconnect disable [ENTER]
```

4. Log out the target port to activate new port settings. Execute:

```
Local> logout {target port no.} [ENTER]
```

5. Activate the macro facility at the target ports. Execute:

```
Local> change port {port no.} macro enable [ENTER]
```

6. Configure those same target ports with an instruction that tells them where to find the specified macro file. Note that the source file may be on either a host processor or a *SmartRAMCARD*. Surround the macro name with quotation marks to retain proper case (upper/lower). Execute:

```
Local> change port {port no.} macro_name  
       /{path to host file}/{filename}.sys"  
       media tftp {IP address of TFTP host} [ENTER]
```

or

```
Local> change port {port no.} macro_name  
       "{filename}.sys" media pc_card [ENTER]
```

For information about variables, refer to appendix A.

Note: When you configure several server ports to execute the same macro filename, the macro file is loaded *once* and that single copy of the macro file is shared by all ports. This saves memory space in the server.

7. Test the newly configured ports. Log into each port and determine if the macro is activated at that port.

Manually Loading a Macro

You can load a macro into the server either from a *SmartRAMCARD* or from a network host.

Downloading the ASCII file converts it to the binary format in which it resides in the server. When the file is converted to binary, it cannot be edited. Therefore, if subsequent editing is required, edit the original ASCII file, and then reload that edited file into the server.

If you limit the size of a macro file to 4K, the server macro facility can store up to eight macro files.

To load a macro from a *SmartRAMCARD* into the server, execute:

```
Local> read macro filename "{filename}.sys" media  
pc_card [ENTER]
```

To load a macro from a network host into the server, execute:

```
Local> read macro filename/{path to host file}  
/"{filename}.sys" media tftp {IP address of  
host} [ENTER]
```

where:

/PATH TO HOST FILE/FILENAME—Is a string of up to 30 alphanumeric characters identifying both the path to the target file containing the macro, and the filename of the macro. The filename is a maximum of 12 alphanumeric characters. The balance of the string is a path. For example, if the filename string is 12 characters long, 18 characters remain for the path string. The complete path includes both the main drive containing the file and the hierarchy of directories or subdirectories through which one must pass to access the target file (e.g., /maindrive/subdir1/subdir2). If the file is located on a host processor, and your using TFTP to transfer the file to the server, you must include the complete path to the source filename.

IP ADDRESS OF HOST—Only applies to file transfers via TFTP. Identifies the internet address of the target TFTP host containing the macro file.

Manually Executing a Macro

When you have loaded a macro into the server, the server macro facility stores the program until it is executed.

1. You must be a privileged user to execute a macro. Change your port to privileged status. Execute:

```
Local> set privileged [ENTER]  
  
Password> {privileged password} [ENTER]
```

2. There is one command to execute a macro:

```
Local> execute macro {macro name} [ENTER]
```

where:

MACRO NAME—Specifies name of file in server containing macro. The specified macro is executed *only* at the user's port.

Displaying Macros

Macros are stored in the server's temporary data base. To produce a listing of the macros in the server, execute:

```
Local> show macros [ENTER]
```

```
Macros currently loaded:
```

```
Macro 1
```

```
Macro 2
```

The following screen is produced:

Removing Macros

You cannot edit macros that are stored in the server. Therefore, if you want to add new features to an existing macro, you must first clear that macro from the server. Then, you can edit the original ASCII file and load that edited file into the server as a new macro. To remove one stored macro, execute the following command specifying the macro name.

```
Local> clear macro {macro name} [ENTER]
```

To remove all stored macros, execute the following command:

```
Local> clear macros local [ENTER]
```

Comparing Macro Facility with VSM

The Macro facility's ability to manage and reconfigure server parameters makes it appear similar to the Vista Server Management (VSM) server management program; however, they are quite different. For example, VSM is a program that runs externally to the server, while the macro facility is part of the server software. Using VSM, a user must configure the *entire* server. Thus, VSM is well-suited to backing up the server configuration so that it can be reloaded in the event of server failure. By contrast, you can create a macro that modifies just a few server parameters. If you create several different macros, each of which modifies different server parameters, and you load those macros into a single server, that server can support several different configurations. The macro facility also can perform functions other than server configuration, such as connecting to a target host.

Storing Configuration Parameters

SmartRAMCARD is a simple, convenient way to manage Communications Server configuration files. *SmartRAMCARD* is a portable memory card which plugs into the reader in an ENIC-equipped Communications Server and either saves or loads configuration files, according to your needs. Using a few simple commands and a *SmartRAMCARD* you can:

- Create a library of server and line card configuration backup files
- Copy existing server and line card configuration settings to other servers
- Develop new configuration settings off-line for later downloading to target servers and intelligent line cards

Configuration Storage and Retrieval

SmartRAMCARD/S manages two types of Communications Server configuration files:

1. **ENIC configuration file**, which is an image of the data stored in the NVRAM portion of the server memory. The server configuration contains:

- Parameters assigned to server-wide characteristics; that is, characteristics affecting all server ports
- Parameters custom-tailored to individual server ports
- Parameters enabling access to network services available through local server ports

2. **“Intelligent” line card configuration file**, which is an image of the data base of user-configurable operating parameters stored in NVRAM-equipped line cards (such as the X.25 PAD card and the 3270 Cluster Controller line card), which are installed in Communications Servers. The line card configuration contains:

- Parameters affecting all users who log into the line card
- Parameters only affecting users of customized line card profiles
- Parameters enabling access to network services accessible through the line card

These user-configurable parameters are text strings (such as service names, passwords and identifiers), numerical values (such as timer limits or group numbers) and switch settings (such as enabled/disabled) that customize the operation of the terminal server to the requirements of the installation site.

As delivered, the server factory default configuration values often accommodate the operating needs of many installations, especially those where no local services are offered. However, where local services are supported, both server and port parameters must be modified. Additional modifications may be executed during server runtime either by port users or by the server manager to improve server throughput in busy networks. The emerging functional configuration, which fine-tunes a server to the demands of both users and the local network, is a composite of parameters developed through field use and repeated experimentation.

Load Slot/save Slot

The **Load Slot** and **Save Slot** commands are closely related and are described together under one heading for convenience. The Load Slot function copies the configuration parameters from a *SmartRAMCARD* to an “intelligent” line card or an ENIC. The save slot function copies the configuration parameters from either an “intelligent” line card or an ENIC to a *SmartRAMCARD/S*. (See figure 4-1.)

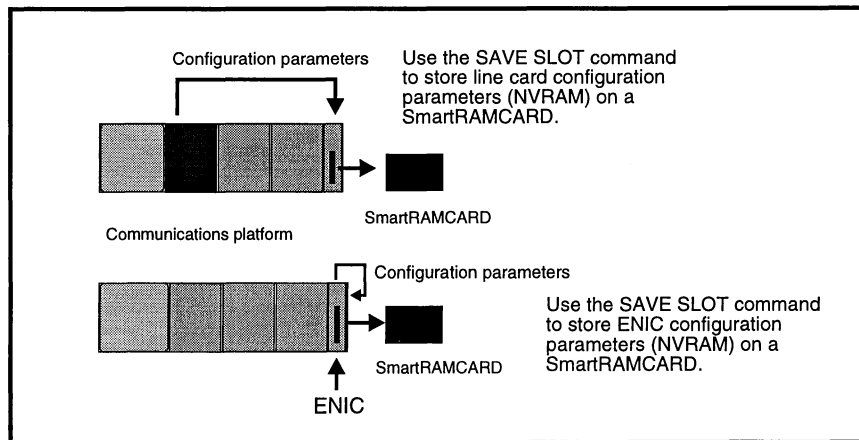


Figure 4-1. Saving NVRAM with SMAR/AMCARD/S

The Communications Server chassis contains five physical slots designed to accommodate both an ENIC and up to four removable line cards. Slots are numbered from zero to four. Slot zero is dedicated to the ENIC. Slots one through four are assigned to line cards. A slot number, included in the **Load** and **Save** commands, identifies the location of the target configuration file.

Execute a **Show Configuration** command to display the locations of the line cards in the server chassis. Then, execute the **Load** or **Save** command specifying the slot number of the target line card.

Each configuration file is identified with a unique filename consisting of a user-selectable eight-character ASCII string. The *SmartRAMCARD/S* automatically appends a three-character suffix to the filename, according to the source of the configuration file. See table 4-3 below.

For example, if the source of the file is an X.25 PAD card, the suffix is .X25. Thus, a complete filename of an X.25 PAD card installed in a server named *server01*, could be *SERVER01.X25*.

More than one configuration file can be stored on a single *SmartRAMCARD/S*.

To load a configuration from the *SmartRAMCARD/S* either to an “intelligent” line card or to an ENIC, specify both the destination slot and the filename from which the configuration parameters are to be taken.

Upon loading a configuration file into an ENIC, the server reboots with the new configuration settings.

Table 4-3. Configuration File: Slot Numbers and File Extensions

Description of configuration file	Server slot number	File extension
Communications Server ENIC configuration	0	.NIC
Communications Server X.25 PAD card configuration	1, 2, 3, or 4	.x25
Communications Server 3270 line card configuration	1, 2, 3, or 4	.327

Using *SmartRAMCARD/S*

Following are procedures to store and retrieve ENIC and “intelligent” line card configurations, and to manage and edit files using the *SmartRAMCARD/S*. Procedures are outlined for each of the following:

- Communications Server
- X.25 PAD card
- 3270 Cluster Controller line card
- Managing files

Saving the Server Configuration

1. Insert the *SmartRAMCARD/S* in the ENIC reader.

Note: If the target server has been booted using a *SmartRAMCARD/F*, remove that card from the reader and replace it with the *SmartRAMCARD/S* that will record the server configuration file. After the configuration is recorded, replace the *SmartRAMCARD/F* in the reader.

2. Determine the current server slot configuration. Confirm the line card locations and their slot assignments in the Communications Server chassis. The ENIC is assigned slot zero (although the ENIC slot assignment is not displayed on the configuration screen). To confirm that the ENIC is in its slot, execute:

```
Local> show configuration [ENTER]
```

3. If the ENIC is properly installed, copy the configuration image to the *SmartRAMCARD/S* in the reader. Execute:

```
Local> save slot 0 {name of file}.nic [ENTER]
```

where:

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file into which the configuration image is stored.

Note: When you specify a filename, the server automatically appends the .NIC suffix to that filename. If you specify a different suffix, the server overwrites that suffix with .NIC.

Upon saving the configuration file to the *SmartRAMCARD/S*, the server prompts you with:

```
Local -607- PCCard access successful.
```

4. Remove the *SmartRAMCARD/S* from the reader. Label the card with the Ethernet address of the source server and store the card in a safe place.

The *SmartRAMCARD/S* is compatible with standard DOS-based PC Card readers and can be inserted in such a reader and read using standard DOS commands.

5. Replace the *SmartRAMCARD/F*, if one is being used. There is no need to reboot the server.

Loading the Server Configuration

1. Insert the *SmartRAMCARD/S* in the ENIC reader.

If the target server has been booted using a *SmartRAMCARD/F*, remove that card from the reader and replace it with the *SmartRAMCARD/S* that contains the server configuration file.

2. Determine the current server slot configuration. Confirm that the ENIC is activated in its slot in the Communications Server chassis. Execute:

```
Local> show configuration [ENTER]
```

3. If the ENIC is properly installed, copy the configuration image from the *SmartRAMCARD/S* to the server. Execute:

```
Local> load slot 0 {name of file}.nic [ENTER]
```

where:

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file in which the configuration image is stored.

4. After the configuration file is copied into the server, the server prompts you with:

```
Local -607- PCCard access successful.
```

5. Are new port NVRAM parameters needed now for runtime operations? If yes, go to step 6. If no, go to step 7.
6. Log out of the port. When the port is logged in, the new parameters become effective.
7. Are new server NVRAM parameters needed now for runtime operations? If yes, go to step 9. If no, go to step 8.
8. Do not reboot the server. Procedure ends here.
9. Was the server booted using a *SmartRAMCARD/F*? If yes, go to step 10. If no, go to step 11.

10. Insert the *SmartRAMCARD/F* and reboot the server.
11. Reboot the server using internal ROM.

Saving X.25 PAD Card Configuration

1. Insert the *SmartRAMCARD/S* in the ENIC reader.

If the target server has been booted using a *SmartRAMCARD/F*, remove that card from the reader and replace it with the *SmartRAMCARD/S* that will record the X.25 PAD Card configuration file. After the configuration is recorded, replace the *SmartRAMCARD/F* in the reader.

2. Determine the current server slot configuration. Locate the server slot containing the X.25 PAD Card and note the number of that slot. Execute:

```
Local> show configuration [ENTER]
```

3. Copy the configuration image from the PAD Card in the slot to the *SmartRAMCARD/S* in the reader. Execute:

```
Local> save slot {number of slot}{name of file}.x25  
[ENTER]
```

where:

NUMBER OF SLOT—Specifies the chassis slot containing the X.25 PAD Card.

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file in which the configuration image is stored.

Note: When you specify a filename, the server automatically appends the .X25 suffix to that filename. If you specify a different suffix, the server overwrites that suffix with .X25.

The configuration **Save/Load** commands support both 10 mHz and 16 mHz versions of the X.25 PAD Card. It is a good practice to choose a filename such as 10MHZ001.X25, which specifies the type of X.25 PAD Card for which configuration parameters are stored.

Upon saving the configuration file to the *SmartRAMCARD/S*, the server prompts you with:

```
Local -607- PCCard access successful.
```

4. Remove the *SmartRAMCARD/S* from the reader. Label the card with the serial number of the PAD Card and the Ethernet address of the source server in which the PAD Card resides. Store the card in a safe place.

The *SmartRAMCARD/S* is compatible with standard DOS-based PC Card readers and can be inserted in such a reader and read using standard DOS commands.

5. Replace the *SmartRAMCARD/F*, if one is being used. There is no need to reboot the server.

Loading X.25 PAD Card Configuration

1. Insert the *SmartRAMCARD/S* in the ENIC reader.

If the target server has been booted using a *SmartRAMCARD/F*, remove that card from the reader and replace it with the *SmartRAMCARD/S* that contains the PAD card configuration file. After the configuration is copied into the PAD card, *only* the affected chassis slot is rebooted; the server is *not* rebooted. When the reboot is complete, replace the *SmartRAMCARD/F* in the reader.

2. Determine the current server slot configuration. Locate the server slot containing the target X.25 PAD card and note the number of that slot. Execute:

```
Local> show configuration [ENTER]
```

3. Copy the configuration image from the *SmartRAMCARD/S* in the reader to the PAD card in the slot. Execute:

```
Local> load slot {number of slot} {name of file}.x25  
[ENTER]
```

where:

NUMBER OF SLOT—Specifies the chassis slot containing the X.25 PAD card.

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file in which the configuration image is stored.

After the configuration file is copied into the X.25 PAD card, the server prompts you with:

```
Local -607- PCCard access successful.
```

4. The X.25 PAD card reboots with the new configuration. Unlike the ENIC configuration load, which reboots the entire server, when an “intelligent” line card is loaded with new configuration values, only the affected slot reboots. The server is *not* rebooted.
5. Replace the *SmartRAMCARD/F*, if one is being used. There is no need to reboot the server.

Saving 3270 Line Card Configuration

1. Insert the *SmartRAMCARD/S* in the ENIC reader.

If the target server has been booted using a *SmartRAM CARD/F*, remove that card from the reader and replace it with the *SmartRAMCARD/S* that will record the 3270 line card configuration file. After the configuration is recorded, replace the *SmartRAMCARD/F* in the reader.

2. Determine the current server slot configuration. Locate the server slot containing the 3270 Cluster Controller line card and note the number of that slot. Execute:

```
Local> show configuration [ENTER]
```

3. Copy the configuration image from the 3270 line card in the slot to the *SmartRAMCARD/S* in the reader. Execute:

```
Local> save slot {number of slot} {name of file}.327
[ENTER]
```

where:

NUMBER OF SLOT—Specifies the chassis slot containing the 3270 Cluster Controller line card.

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file in which the configuration image is stored.

When you specify a filename, the server automatically appends the .327 suffix to that filename. If you specify a different suffix, the server overwrites that suffix with .327.

Upon saving the configuration file to the *SmartRAMCARD/S*, the server prompts you with:

```
Local -607- PCCard access successful.
```

4. Remove the *SmartRAMCARD/S* from the reader. Label the card with the serial number of the 3270 Cluster Controller line card and the Ethernet address of the source server in which the line card resides. Store the card in a safe place.

The *SmartRAMCARD/S* is compatible with standard DOS-based PC Card readers and can be inserted in such a reader and read using standard DOS commands.

5. Replace the *SmartRAMCARD/F*, if one is being used. There is no need to reboot the server.

Loading 3270 Cluster Controller Configuration

1. Insert the *Smart*RAMCARD/S in the ENIC reader.

If the target server has been booted using a *Smart*RAMCARD/F, remove that card from the reader and replace it with the *Smart*RAMCARD/S that contains the line card configuration file. After the configuration is copied into the line card, *only* the affected chassis slot is rebooted; the server is *not* rebooted. When the reboot is complete, replace the *Smart*RAMCARD/F in the reader.

2. Determine the current server slot configuration. Locate the server slot containing the target 3270 Cluster Controller line card and note the number of that slot. Execute:

```
Local> show configuration [ENTER]
```

3. Copy the configuration image from the *Smart*RAMCARD/S in the reader to the 3270 line card in the slot. Execute:

```
Local> load slot {number of slot} {name of file}.327  
[ENTER]
```

where:

NUMBER OF SLOT—Specifies the chassis slot containing the 3270 Cluster Controller line card.

NAME OF FILE—Specifies a string of up to eight alphanumeric characters, plus a three-character suffix identifying the file in which the configuration image is stored.

After the configuration file is copied into the 3270 Cluster Controller line card, the server prompts you with:

```
Local -607- PCCard access successful.
```

4. The 3270 line card reboots with the new configuration. Unlike the ENIC configuration load, which reboots the entire server, when an “intelligent” line card is loaded with new configuration values, only the affected slot reboots. The server is *not* rebooted.
5. Replace the *Smart*RAMCARD/F, if one is being used. There is no need to reboot the server.

Managing Smart RAMCARD/S Files

Files on the *SmartRAMCARD/S* can be examined and modified using standard DOS commands. Table 4-4 describes three commands that help you manage these files.

The selected file can be opened and modified using line editing commands. For further information, refer to your DOS manual.

Table 4-4. File Management Commands

Command	Description
DIR	This command is equivalent to the DOS DIR command. It lists all the files on the <i>SmartRAMCARD/S</i> .
REN {oldname} {newname}	This command renames the filename from the old name to a new name.
DEL {filename}	This command deletes the filename from the <i>SmartRAMCARD/S</i> .

Storage/Retrieval Status Messages

Status messages are produced during the **Load Slot** and **Save Slot** operations. Each message is described and supported with a procedure for corrective action, if necessary. Table 4-5 lists the status messages with descriptions and required actions.

Table 4-5. Status Messages and Required Actions

Status message	Description	Action required
601 PCCard access successful	The Save or Load operation has completed successfully.	No action required.
602 No Line Card in the slot number	The slot number specified in the command is empty and does not contain any line card.	1. Execute a Show Configuration command and determine the slot number of the target line card. 2. Retry the Save or Load operation using the slot number specified in the display.
603—Unknown line card in slot number	The slot number specified in the command contains a line card other than a 3270 Cluster Controller line card or an X.25 PAD card.	1. Execute a Show Configuration command and determine the slot number of the target line card. 2. Retry the Save or Load operation using the slot number specified in the display.
604—PC card media not found	The reader cannot detect the <i>SmartRAMCARD/S</i> .	1. Make sure <i>SmartRAMCARD/S</i> is correctly inserted and fully seated in the reader. 2. Retry the Save or Load operation.
605—Not enough free space on PC card	The <i>SmartRAMCARD/S</i> is full of data and cannot accept additional characters. The server will not write a partial file to a <i>SmartRAMCARD/S</i> ; only complete files are stored on a card.	1. Remove the filled card and replace it with a new card. or: 1. Insert the filled card in a PC equipped with a reader. 2. Copy files to the PC's hard drive. 3. Delete files from the card to generate free space. 4. Retry the Save procedure. 5. Delete additional files, if necessary.
606—Error writing filename to PC card	The file specified in the Save command could not be taken from the server and written to the <i>SmartRAMCARD/S</i> .	See status message 605.

Status message	Description	Action required
607—Filename not found on PC card	The filename specified in the Load command was not found on the <i>SmartRAMCARD/S</i> .	<ol style="list-style-type: none"> 1. Compare the actual filename with the character string entered in the command. 2. If the filename was incorrectly entered, re-type the command and the filename and re-execute the command. 3. If the filename was correctly entered, insert the <i>SmartRAMCARD/S</i> in a PC equipped with a reader and check for the presence of the file. 4. If the file is not on this <i>SmartRAMCARD/S</i>, check each <i>SmartRAMCARD/S</i> for the file.
608—Error reading filename from PC card	<p>The filename specified in the Load command was found on the <i>SmartRAMCARD/S</i> and was opened, but for some reason, the file could not be read.</p> <p>(The file may have been truncated to less than its full size.)</p>	<ol style="list-style-type: none"> 1. Remove the <i>SmartRAMCARD/S</i> from the reader. 2. Insert the card in a reader-equipped PC. 3. Check the size of the file that was specified in the Load command. Configuration file sizes are adjustable depending upon the number of configured server ports. While the upper limits are variable, any file size less than 60K should be considered a truncated file. 4. If the file is truncated, it cannot be loaded.

Event Logger Messages

The new events log messages are generated during both the configuration storage and retrieval procedures and server software upload and download.

The messages in table 4-6 appear on the console and in the event logger.

Table 4-6. Event Logger Messages

Event number	Message	Description
1600	Storing slot configuration to PCC	When the Save Slot {number} command executes successfully, this message appears indicating that the NVRAM of an "intelligent" line card has been stored in the <i>SmartRAMCARD/S</i> .
1601	Retrieving slot configuration from PCC	When the Load Slot {number} command executes successfully, this message appears indicating that the file containing the contents of the NVRAM that was stored from an "intelligent" line card, now has been copied into a line card.
1602	Storing ENIC configuration to PCC	When the Save Slot 0 command executes successfully, this message appears indicating that the NVRAM of an ENIC has been stored in the <i>SmartRAMCARD/S</i> .
1603	Retrieving NIC configuration from PCC	When the Load Slot 0 command executes successfully, this message appears indicating that the file containing the contents of the NVRAM that was stored from an ENIC, now has been copied into an ENIC.
1604	Slot 1 re-initialized	When the Load Slot 1 command executes successfully, this message appears indicating that the line card was rebooted with the new configuration parameters.
1605	Slot 2 re-initialized	When the Load Slot 2 command executes successfully, this message appears indicating that the line card was rebooted with the new configuration parameters.
1606	Slot 3 re-initialized	When the Load Slot 3 command executes successfully, this message appears indicating that the line card was rebooted with the new configuration parameters.
1607	Slot 4 re-initialized	When the Load Slot 4 command executes successfully, this message appears indicating that the line card was rebooted with the new configuration parameters.

Software Uploading Options

During its normal activities, the server operates using a set of programming instructions (called its *software image*) together with a file of parameter settings (called its *configuration file*). The software image is the conglomeration of all the individual software files and subroutines that comprise the server's operating system. The configuration file contains the variable parameter settings; that is, the name-strings and numerical values that define users, passwords, port numbers, local services, and data flow parameters unique to the installation. The software image only tells the server how to execute commands; it does not tell the server what parameter settings to use.

As shown in figure 4-2, server configuration settings are stored in a secure portion of NVRAM, apart from the server software image, which resides in the server's onboard ROM. Because they are stored in separate memory locations, the server's configuration settings and its software image can be up- and down-loaded independently. For example, if the server's current software image is reloaded, the configuration values stored in that server remain unchanged.

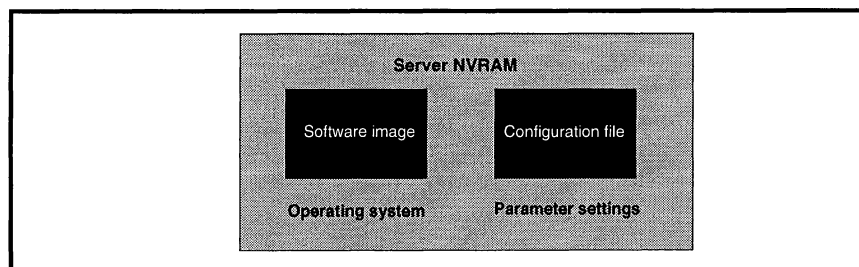


Figure 4-2. Configuration file is stored separately

Caution: If you load an upgraded version of software into a site-configured server, and there are new parser commands in the upgraded version, you risk losing the current configuration file in the target server. Always copy the server's configuration image to a backup medium, such as a *SmartRAMCARD*, or use Penril Datability Networks' VSM product to retrieve configuration values from the server and save them in a file you can edit, if desired. After the new software image is loaded, copy the old configuration file back into the upgraded server.

Obtaining a Software Image

Each time it boots, the server obtains a copy of its software image from a designated source. First, the server tries (for up to 3 minutes) to retrieve a software image from its designated primary boot source. If the primary source does not respond within the allocated time period, the server switches to its alternate (secondary) source and repeats the procedure.

Software images are available from a variety of sources. The server retains an onboard software image in its Read Only Memory (ROM). Alternative sources include LAT-based hosts, UNIX-based hosts, another server or a *SmartRAMCARD*.

The operating software for newer versions of stand-alone servers is not compatible with the software in earlier versions of servers. Filenames for earlier versions begin with the prefix *DSSR*, while newer versions begin with the prefix *EM2R*.

Primary and Secondary Boot Sources

The server factory default configuration sets the primary booting source to *SmartRAMCARD* and the secondary booting source to ROM. To change these settings, follow the steps below.

1. Log into a server port and set that port to privileged operation. Execute:

```
Local> set privileged [ENTER]
```

2. Supply the privileged password. Execute:

```
Password {privileged password string} [ENTER]
```

3. Activate the server Autoboot feature. Execute:

```
Local> change server autoboot enabled [ENTER]
```

4. When Server Autoboot is enabled, you can designate both primary and secondary sources from which the server can retrieve a software image. To assign a primary source, execute:

```
Local> change server primary boot {boot source}  
[ENTER]
```

Where `boot source` is one of the following variables:

ROM—Instructs the server to boot from the software image stored in its Read-only Memory.

MOP—Instructs the server to use the Maintenance Oriented Protocol to obtain a software image from either a LAT-based host or a VCP server configured for software export.

TFTP—Instructs the server to use the Trivial File Transfer Protocol to obtain a software image from either a LAT-based or a UNIX-based network host, or a VCP server configured for software export.

BOOTP—Instructs the server to use the bootstrap protocol to obtain a software image from a UNIX-based network host.

CARD—Instructs the server to retrieve the software image stored on the *SmartRAMCARD* inserted in the server card reader. If you do not select a primary boot source, the server factory default is Card. If Card is the primary boot source and there is no *SmartRAMCARD* in the card reader when the server is booted, it will attempt to retrieve a software image from its secondary source.

5. To assign a secondary source, execute:

```
Local> change server secondary boot {boot source}
[ENTER]
```

Where *boot source* is one of the following variables:

NONE—Specifies that there is no secondary software image source. The server must have a primary boot source; however, a secondary boot source is an option. Therefore, one of the choices is NONE.

ROM, MOP, TFTP, BOOTP, CARD—For descriptions of these boot sources, see the list of primary boot sources in step 4. If you do not select a *secondary* server boot source, the factory default is server internal ROM.

Software Images Transferred Via Network

Server software images can be copied (or uploaded) from a source device to a destination server. The source device can be a host processor (such as a VAX running VMS or UNIX operating systems), another server or a *SmartRAMCARD*. Files are transferred using one of three available protocols:

- Maintenance Oriented Protocol (MOP)
- Trivial File Transfer Protocol (TFTP)
- Bootstrap Protocol (BOOTP)

VAX/VMS hosts can use any of these three protocols to transfer files; however, UNIX hosts are restricted to TFTP and BOOTP. Note that BOOTP cannot be used to transfer files between servers or from a *SmartRAMCARD* to a server. In those instances, transfers are limited to MOP and TFTP.

Table 4-7 illustrates the available server software image transfer options. Each of these options is discussed further in the procedures that follow.

Table 4-7. Software Transferring Image Transfer Options

Source		Destination		File transfer program		
				MOP	TFTP	BOOTP
Case 1: host to server	Host	Local network	Server 1 Executing Image	VAX/ VMS host only	VAX/ VMS host	VAX/ VMS host
					UNIX host	UNIX host
Case 2: server to server	Server 1 Executing Image	Local network	Server 2 Executing Image	Yes	Yes	Not applicable
Case 3: Smart RAMcard to server	Server 1 Smart RAM card	Local network	Server 2 Executing Image	Yes	Yes	Not applicable

Uploading from a Host to a Server Using MOP

MOP is used to transfer software images quickly between a source host and a target server. MOP works with LAT protocol. MOP works quickly because it transfers data in 1000 byte increments, rather than the 512 byte increments used in slower protocols, such as TFTP and BOOTP. In addition to its advantage of speed, MOP also generates useful status messages during the software transfer process.

To upload server software from a VAX/VMS source host, execute the following procedure:

1. Obtain a tape containing the software image to be uploaded to the target server. For further information about obtaining software image tapes, contact your service representative.
2. Install the EM2RV001.SYS file into the MOP directory on the source host.

Note: The software image file, EM2RV001.SYS, can be installed on a PC and copied from that PC to a target server via a file transfer utility.

3. At the target server, set the primary boot source to MOP. Execute:

```
Local> change server primary boot mop [ENTER]
```

4. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

5. Add, to the target server, the name of the software file containing the server software on the host. Execute:

```
Local> change server software em2rv001 [ENTER]
```

6. Upload the software image from the host to the server by rebooting the server. Execute:

```
Local> crash [ENTER]
```

Turning the server off, then on, is another way to reboot it.

7. When the server reboots it loads the software from the source host.

Uploading from a Host to a Server Using TFTP

TFTP is used to transfer software images between either VAX/VMS or UNIX hosts and a target (or destination) server. TFTP processes packets at a slightly slower pace than MOP. Thus, if the host is a VAX/VMS processor, MOP is the preferred protocol. If the host is a UNIX-based processor, MOP cannot be used, and either TFTP or BOOTP (discussed later) are the available protocols.

To upload server software from a UNIX-based source, follow these directions:

1. Obtain a tape containing the server software image to be uploaded to the target server. For further information about obtaining software image tapes, contact your service representative.
2. Install the EM2RV001.SYS file into the uploading UNIX host. Locate the IP address of the uploading host, and copy the file into the appropriate directory (such as PATH_NAME).
3. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

4. At the target server, set the primary boot source to TFTP. Execute:

```
Local> change server primary boot tftp [ENTER]
```

5. Assign an IP address to the target server. Execute:

```
Local> change address {IP address of target server}  
[ENTER]
```

6. At the target server, assign the name of the load host boot file containing the software image. Execute:

```
Local> change server software em2rv001 [ENTER]
```

7. At the target server, assign the IP address of the uploading UNIX host; that is, the server bootserver. Execute:

```
Local> change server bootserver address  
      {IP address of uploading host} [ENTER]
```

8. At the target server, assign a path of directories pointing to the UNIX host directory containing the software image file. Execute:

```
Local> change server path /{path to image file}/  
      [ENTER]
```

Note: Include the virgules (/) in this command.

9. Upload the software image from the host to the target server by rebooting the server. Execute:

```
Local> crash [ENTER]
```

Note: Turning the server off, then on, is another way to reboot it.

10. When the server reboots, it loads an executing image from the source host.

Uploading from a Host to a Server Using BOOTP

BOOTP is used to transfer software images between either VAX/VMS or UNIX hosts and a target (or destination) server. BOOTP processes packets at a slightly slower pace than MOP. Thus, if the host is a VAX/VMS processor, MOP is the preferred protocol. If the host is a UNIX-based processor, MOP cannot be used, and either BOOTP or TFTP (discussed above) are the available protocols.

Use BOOTP when the target server has an Ethernet address but does not have an IP address. In this case, when the target server is rebooted, it sends an Ethernet broadcast frame containing a request for reverse address resolution. This, so-called, RARP packet is received by a network host which searches its own data base for both the Ethernet address and the corresponding IP address of the target server. The responding host returns (to the target server), the IP address, the server bootserver IP address, the server pathname (or, home directory), and the name of the image bootfile. Upon receiving this information, the target server executes a TFTP process to complete the upload.

To upload server software from a UNIX-based host, execute the following:

1. Obtain a tape containing the software image to be uploaded to the target server. For further information about obtaining software image tapes, contact your service representative.

2. At the uploading (BOOTP) host, modify the *BOOTPTAB* file (in the */ETC* directory) with the path directory name and the server software filename (*EM2RV001.SYS*), IP address, and corresponding Ethernet address of the target server, as follows:

```

{server hostname}

:sm={IP broadcast address}

:had={path directory}:bf={name of software image}:\

:ht={hardware type}:ha={ethernet address of target
server}:\

:ip={IP address of target server}:ts={IP address of
uploading bootserver}:\

:ds={IP address of TFTP host} {IP address of upload-
ing bootserver}:\

:to={timeout value in milliseconds}:

```

For example, consider the configuration required to upload software from a UNIX host (IP address 123.45.6.2) to the server, host.corp.com, at IP address 123.45.6.180, with bootserver IP address 123.45.6.33, and Ethernet address 01-02-03-a4-b5-c6. The path directory is */TFTPBOOT* and the server software filename is *EM2RV001.SYS*. Using a timeout value of 18 seconds, one might execute this procedure at the UNIX host as follows:

```

host.corp.com

:sm=255.255.255.0

:had=/tftpboot:bf=em2rv001.sys:\

:ht=ethernet:ha=010203a4b5c6:\

:ip=123.45.6.180:ts=123.45.6.33:\

:ds=123.45.6.2 123.45.6.33:\

:to=18000:

```

Note: BOOTP does not require the target server to have an IP address. Only the server Ethernet address is needed for the upload.

3. Install the *EM2RV001.SYS* file into the path directory specified in the *BOOTPTAB* file on the uploading UNIX host.

4. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

5. At the target server, set the primary boot source to BOOTP. Execute:

```
Local> change server primary boot bootp [ENTER]
```

6. At the target server, assign the IP address of the local server. Execute:

```
Local> change ip {IP address of target server} [ENTER]
```

7. Upload the software image from the host to the target server. Execute:

```
Local> crash [ENTER]
```

Note: Turning the server off, then on, is another way to reboot it.

8. When the target server reboots, it loads an executing image from the source host.

Uploading from Server to Server Using MOP

MOP is used to transfer software images quickly between a source server and a target server. Both LAT and Telnet are supported by MOP. The source server maintains an executing image which it uploads to the target server.

Specific filenames are indicated in the file transfer procedure. The filename for the Communications Server is EM2RV001. This filename *must* be used, or the target server will halt the upload process.

To upload the software image from a source server to a target server, execute the following procedure:

1. At the source server, activate the software export function. Execute:

```
Local> change server export enable [ENTER]
```

2. At the target server, set the primary boot source to MOP. Execute:

```
Local> change server primary boot mop [ENTER]
```

3. At the target server, activate the Autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

4. Add, to the target server, the name of the software file containing the server software on the source server. Execute:

```
Local> change server software em2rv001 [ENTER]
```

Note: If the source server has been uploaded with a software image from a *SmartRAMCARD*, that card must remain inserted in the reader of that source server at all times. If a power outage or system abnormality causes the source server to reboot, the desired software image is in the *SmartRAMCARD*. Subsequent uploads to target servers also use the *SmartRAMCARD* software image. For further information see *Uploading from SmartRAMCARD to Server*.

5. Upload the software image from the source server to the target server. Execute:

```
Local> crash [ENTER]
```

Turning the server off, then on, is another way to reboot it.

6. When the target server reboots it loads the executing image of the source server.

Uploading from Server to Server Using TFTP

TFTP is used to transfer software images from a source server to a target server. Recall that TFTP file transfers are somewhat slower than MOP file transfers, due to the maximum allowable TFTP packet size (512 bytes for TFTP vs. 1000 bytes for MOP). Thus, MOP is a preferred file transfer protocol.

Specific filenames are indicated in the file transfer procedure. The filename for the Communications Server is *EM2RV001*. This filename *must* be used, or the target server will halt the upload process.

To upload the software image from a source server to a target server, execute the following procedure:

1. Assign an IP address to the source server. Execute:

```
Local> change address {IP address of source server}  
[ENTER]
```

2. At the source server, activate the server software export feature. Execute:

```
Local> change server export enable [ENTER]
```

3. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

4. At the target server, set the primary boot source to TFTP. Execute:

```
Local> change server primary boot tftp [ENTER]
```

5. Assign an IP address to the target server. Execute:

```
Local> change address {IP address of target server}  
[ENTER]
```

6. At the target server, assign the IP address of the uploading server bootserver. Execute:

```
Local> change server bootserver {IP address of  
source server} [ENTER]
```

7. At the target server, assign the name of the software file containing the server software on the source server. Execute:

```
Local> change server software em2rv001 [ENTER]
```

Note: If the source server was uploaded with a software image from the *SmartRAMCARD*, that card *must* remain inserted in the source server at all times. If a power outage or system abnormality causes the source server to reboot, the desired software image is in the *SmartRAMCARD*. Subsequent uploads to target servers also use the *SmartRAMCARD* software image, as detailed in the section, *Uploading from SmartRAMCARD to Server*.

8. Upload the software image from the source server to the target server. Execute:

```
Local> crash [ENTER]
```

Turning the server off, then on, is another way to reboot it.

9. When the target server reboots, it loads an executing image from the source server.

Uploading from SmartRAMCARD to Server

The available portable *SmartRAMCARD* device stores upgraded software that can be read into the server. *SmartRAMCARD* uploading is an attractive alternative to traditional methods of software distribution and upgrading.

The *SmartRAMCARD* is made of stiff plastic, measuring 3.375 inches by 2.0 inches by 0.125 inches. This is roughly the footprint of a stack of two or three credit cards. At the forward end of the card is a female 64-pin connector. The card is inserted for reading (forward end first), through a slot in a panel in the server. Graphics printed on the face identify the forward end of the card.

Although *SmartRAMCARDS* are fairly durable, they do have a few modest physical requirements. *SmartRAMCARDS* should be kept:

- Dry
- Away from extreme heat or cold (do not lay them on top of your computer or store them on the seat of your car)
- Out of direct sunlight

Note: Uploading from a *SmartRAMCARD* to a server can take up to 5 minutes, due to the compressed nature of the software on the card.

Uploading From a Local Server

You can upload server software from a *SmartRAMCARD* to a local server as follows:

1. Hold the *SmartRAMCARD* so that the female 64-pin connector is facing the reader slot.
2. Insert the *SmartRAMCARD* into the reader slot. Apply gentle but firm pressure to the card until it is fully seated in the reader connector socket. When the card is fully inserted, the **Eject** button, located at the bottom of the slot, is fully extended.
3. Return the server software to its factory default configuration settings. Execute a reboot:

```
Local> reset [ENTER]
```

4. The server prompts you with:
5. Press **Y** to return the server to its factory default settings.
6. When the target server reboots, the *SmartRAMCARD* software is loaded into the local server and becomes that server's executing image.
7. Do not remove the *SmartRAMCARD* from the server. The card must remain in the server as a dedicated source of software, which is available if a power outage or other abnormality occurs and the server must be rebooted.
8. If the *SmartRAMCARD* must be removed, press the **Eject** button at the reader. Do *not* attempt to remove the *SmartRAMCARD* from the unit without first pressing the **Eject** button.

Uploading from SmartRAMCARD to Server via MOP

The *SmartRAMCARD* software image file can be transferred from the source server (in which it resides) to remote servers connected to the local area network. To upload the software image from a *SmartRAMCARD* to a target server, execute the following procedure:

1. At the source server, if the executing image was uploaded from a *SmartRAMCARD*, insert that *SmartRAMCARD* into the reader. If not; that is, if the executing image resides in server ROM, go to step 5.
2. See *Uploading From a Local server*.
3. At the source server, activate the export function. Execute:

```
Local> change server export enable [ENTER]
```

4. Remove the *SmartRAMCARD* from the source server.
5. At the source server, insert the *SmartRAMCARD* containing the software image required at the target server.

Note: *SmartRAMCARD* software can be different from the software image in the source server. This feature permits one to insert a card (into a server) containing the software image of one server and upload that image to a remote server. While this feature is provided, it probably is of dubious value since the card *must* remain in the source server for subsequent uploads to the target server. Obviously, the image in the dissimilar card cannot be written to the executing image area of the source server. When the source server is booted, the dissimilar card *must be removed* and exchanged for the proper card, which must be inserted to upload the correct software image.

6. At the target server, set the primary boot source to MOP. Execute:

```
Local> change server primary boot MOP [ENTER]
```

7. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```

8. Add, to the target server, the name of the software file containing the server software on the card. Execute:

```
Local> change server software em2rv001 [ENTER]
```

Note: If the source server has been uploaded with a software image from a *SmartRAMCARD*, that card must remain inserted in the reader of that source server at all times. If a power outage or system abnormality causes the source server to reboot, the desired software image is in the *SmartRAMCARD*. Subsequent uploads to target servers also use the *SmartRAMCARD* software image. For further information see *Uploading from SmartRAMCARD to Server*.

9. Upload the software image from the source server to the target server. Execute:

```
Local> crash [ENTER]
```

Turning the server off, then on, is another way to reboot it.

10. When the target server reboots it loads an executing image from the *SmartRAMCARD* in the source server.

Uploading to Remote Server via TFTP

The *SmartRAMCARD* software image file can be transferred from the source server (in which it resides) to remote servers connected to the LAN by using TFTP. To upload the software image from a *SmartRAMCARD* to a target server, execute the following procedure:

1. At the source server, if the executing image was uploaded from a *SmartRAMCARD*, insert that *SmartRAMCARD* into the server. If not; that is, if the executing image resides in the server ROM, go to step 7.
2. See *Uploading From a Local Server*.
3. Assign a local IP address to the source server. Execute:

```
Local> change address {IP address of source server}
[ENTER]
```
4. At the source server, activate the export function. Execute:

```
Local> change server export enable [ENTER]
```
5. Remove the *SmartRAMCARD* from the source server.
6. At the source server, insert the *SmartRAMCARD* containing the software image required at the *target* server.
7. Assign a local IP address to the target server. Execute:

```
Local> change address {IP address of target server}
[ENTER]
```
8. At the target server, set the primary boot source to TFTP. Execute:

```
Local> change server primary boot tftp [ENTER]
```
9. At the target server, activate the autoboot function. Execute:

```
Local> change server autoboot enable [ENTER]
```
10. Add, to the target server, the name of the software file containing the server software on the *SmartRAMCARD*. Execute:

```
Local> change server software em2rv001 [ENTER]
```

Note: If the source server has been uploaded with a software image from a *SmartRAMCARD*, that card must remain inserted in that source server at all times. If a power outage or system abnormality causes the source server to reboot, the desired software image is in the *SmartRAMCARD*. Subsequent uploads to target servers also use the card software image. For further information see *Uploading from SmartRAMCARD to Server*.

11. Upload the software image from the *SmartRAMCARD* in the source server to the target server. Execute:

```
Local> crash [ENTER]
```

Turning the server off, then on, is another way to reboot it

12. When the target server reboots it loads an executing image from the *SmartRAMCARD* in the source server.

Managing the Server Timeserver

The server maintains an internal time manager that controls its calendar and clock functions. The calendar and clock are used to timestamp eventlog messages. They also can be used to set the clocks of other hosts on the network. In addition to a date and time, a time zone also can be specified. The time zone parameter is significant if the server will be connecting to hosts in locations around the world.

Time and date management functions include:

- Setting the time manager to the correct date and time
- Assigning time zones
- Controlling operation of the server as a network timeserver

Setting the Current Time and Date

To set the time and date in the server time manager, proceed as follows:

The time and date only can be set; they *cannot* be changed or defined.

1. Enter the current time of day using 24-hour time notation in hour:minute:second format, where: hh is 00 to 23; mm is 00 to 59; ss is 00 to 59. Execute:

```
Local> set time {hh:mm:ss} [ENTER]
```

2. Enter the current date in date-month-year format, where: dd is 01 to 31; mm is 01 to 12; and yy is 00 to 99. Execute:

```
Local> set time {dd-mm-yy} [ENTER]
```

The time and date also can be entered in a single command as follows:

```
Local> set time {hh:mm:ss} {dd-mm-yy} [ENTER]
```

- 3a. To have the server automatically request the current time and date from the network timeserver upon start-up, execute the auto command:

```
Local> change time auto [ENTER]
```

- 3b. If you want the server to request the current time and date upon rebooting, you must choose the protocol the server will use to make the request (Broadcast UDP, IP, or MOP). If you do not select a protocol, the server automatically uses MOP. Execute:

```
Local> change time {broadcast} {IP} {MOP} [ENTER]
```

The Time Auto and Time protocol commands only can be changed or defined. They cannot be set.

- 3c. To instruct the server to automatically reset itself to network time at the same time each day, enter the Resync parameter immediately following the network protocol parameter. For example, to instruct the server to use IP to synchronize its internal clock/calendar every day at 6:00 PM, execute the following command:

```
Local> change time ip resynch 18 [ENTER]
```

Geographic Time Zones

The internal time manager function allows you to specify a source (either North American geographic location or Greenwich Mean Time) of the current date/time setting in the server. For servers that will be connecting to hosts in other cities, possibly across one or more time zones, setting the internal timeserver to a specific geographic time zone ensures accuracy of the internal clock.

For example, if the local server is set to E.S.T. (Eastern Standard Time) and, if another host, such as a remote server, is a network timeserver, the local server will resynchronize its internal clock to that of the remote timeserver. If the remote timeserver is in a different time zone, such as C.S.T. (Central Standard Time), when the local server receives a timecheck (such as 07:30:00) from that timeserver, it resets itself to the remote time and then adjusts its clock for the difference between the time zones. Thus, the local server would set its clock to 08:30:00, E.S.T. Because the time zone adjustment process takes a few seconds, the clocks may not be synchronized to the exact second.

Setting Standard Time Zone

To set the standard time zone in your server, decide whether your server time manager will synchronize with a North American time zone or with Greenwich Mean Time. Execute:

```
Local> change timezone standard {North American  
time zone} [ENTER]
```

or Local> **change timezone standard gmt_off +{h}** [ENTER]

or

```
Local> change timezone standard gmt_off -{h} [ENTER]
```

Where:

NORTH AMERICAN TIME ZONE—Is EST, CST, MST, or PST.

GMT—Synchronizes the time manager with Greenwich Mean Time.

GMT_OFF +h—Indicates the number of hours to add to Greenwich Mean Time to synchronize the server's internal clock with your local time zone.

GMT_OFF -h—Indicates the number of hours to subtract from Greenwich Mean Time to synchronize the server's internal clock with your local time zone.

Setting Alternate Time Zone

You can program the server to change from its standard time zone to an alternate time zone. To set the alternate time zone in your server, decide whether your server time manager will synchronize with a North American time zone or with Greenwich Mean Time. Execute:

```
Local> change timezone alternate {North American  
timezone} [ENTER]
```

or

```
Local> change timezone alternate gmt_off +{h} [ENTER]
```

or

```
Local> change timezone alternate gmt_off -{h} [ENTER]
```

Where:

NORTH AMERICAN TIME ZONE—Is EDT, CDT, MDT or PDT.

GMT_OFF—Synchronizes the time manager with Greenwich Mean Time.

GMT_OFF +h/GMT_OFF -h—See above.

Switching Between Standard and Alternate Time Zones

You can program the server time manager to automatically switch itself from a standard time zone to an alternate time zone and back again on specified dates. This is useful at those points in the year when a time zone changes from Standard Time to Daylight Saving Time, and vice-versa.

To set a switch date between standard and alternate time zones, proceed as follows:

1. Specify the month and date at which you want the server time manager to change its clock from its standard time zone setting to an alternate time zone setting. Execute:

```
Local> change timezone change {dd-mm} [ENTER]
```

Where:

DD—Specifies the digits of the date (01-31).

MM—Specifies the digits of the month (01-12).

2. Specify the month and date at which you want the server time manager to change its clock from its alternate time zone setting to a standard time zone setting. Execute:

```
Local> change timezone return {dd-mm} [ENTER]
```

Enabling or Disabling the Internal Timeserver

The server Timeserver can be set to either respond to or ignore data and time requests from other devices on the network. The Timeserver commands control the server's ability to respond to these network date and time requests. To alter this parameter, proceed as follows:

1. To enable the server to respond to network date/time requests, execute:

```
Local> set timeserver enable [ENTER]
```

The factory default setting is Enabled.

2. To prevent the server from responding to network date and time requests, execute the following command:

```
Local> set timeserver disable [ENTER]
```

Tracing a Route on the Network

Traceroute is a diagnostics tool that traces the route a packet takes from the server to a target host. If you have difficulty connecting to a target host you can execute this command and it will generate a listing of the IP addresses of the intermediate routers and the delays encountered at each hop along the route to the target host.

When its TTL value equals number of hops from source server, query packet “dies” and gateway sends ICMP message back to source. ICMP message issued by gateway where query packet *died* contains the IP address of that gateway and the roundtrip travel time between the source server and that gateway. The sum of the incremental roundtrip times equals the total roundtrip time from the source server to the target host. See figure 4-3.

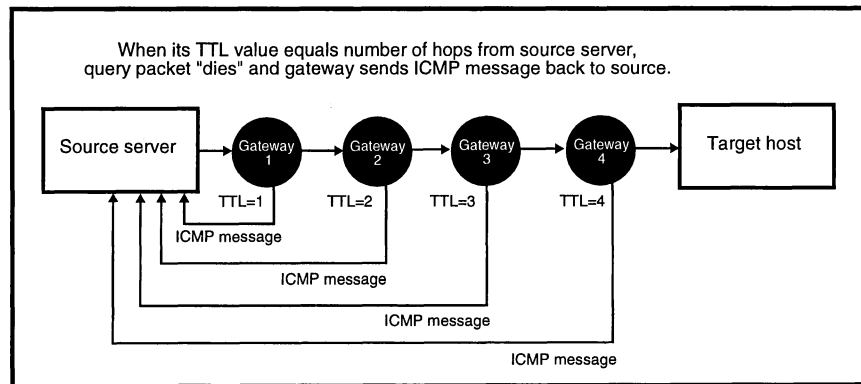


Figure 4-3. TTL values of ICMP messages

Setting UDP Port Parameter

Traceroute packets are sent to the UDP port of the target host. The server does not want to connect to the UDP port, but it does want to receive ICMP packets from it. So, it picks a target UDP port that no application uses and sends its probes there.

However, if by chance, an application is using the target UDP port, you can change the port number by executing:

```
Local> change traceroute_db udpport {target UDP  
port number} [ENTER]
```

where:

TARGET UDP PORT NUMBER—Is an integer from 1 to 65535. The factory default is 30000.

Setting Query Parameter

By default, traceroute sends a series of *three* query packets, each with the same TTL value, from the source server to the target host. First, traceroute dispatches a query packet with a TTL value of *one*. When the source server gets a response to that query, it sends a second query packet with a TTL of *one* and waits for a response. After that response is received, the server sends a third query packet with a TTL of *one* and waits for a response. These responses are recorded in the traceroute data base and are displayed as traceroute output.

After the query limit is reached, traceroute increments the TTL counter by one and repeats the process by sending another batch of query packets to the target. These query packets will travel a little further—*one more hop*—from the source server before they die. More statistics are accumulated as each new batch of query packets dies at a new gateway. The TTL is incremented either until the target host responds or until the maximum TTL value is reached.

Dispatching multiple query packets allows you to calculate an average of the roundtrip time delay values recorded at each hop. You can change this parameter to an integer from one to 16. To adjust this parameter, execute:

```
Local> change traceroute_db query {no. of queries}  
[ENTER]
```

Setting MAXTTL Parameter

By default, traceroute limits the life of a query packet to 30 hops. You can change this parameter to an integer from one to 255. To adjust this parameter, execute:

```
Local> change traceroute_db maxttl {maximum no. of  
hops} [ENTER]
```

Setting Timeout Parameter

By default, traceroute waits 3 seconds for a response from a query packet. You can change this parameter to an integer from one to 255. To adjust this parameter, execute:

```
Local> change traceroute_db timeout {no. of seconds}  
[ENTER]
```

Setting Packet Size Parameter

By default, traceroute limits the size of a query packet to 56 bytes. You can change this parameter to an integer from one to 512. Bigger packets may help detect packet fragmentation, which can slow the transmission process. Bigger packets may be sent over different routes than smaller packets. To adjust this parameter, execute:

```
Local> change traceroute_db packet_size {no. of  
bytes} [ENTER]
```

Starting Traceroute

To start sending traceroute query packets to the target host, execute:

```
Local> traceroute {target host domain} {source IP  
address} [ENTER]
```

where: **TARGET HOST DOMAIN**—Specify the domain name of the target host. When the command is executed, traceroute will call the nameserver to resolve the domain into an IP address.

SOURCE IP ADDRESS (*Optional*)—If the local server has several IP addresses, you can execute the command using the IP address that will originate the query packets. If you do not specify the source IP address, the ICMP packets are returned to the server's main IP address.

Example of Traceroute Output

Refer to the following example below of a traceroute inquiry. The target host is the domain `nis.nsf.net`, which the nameserver resolved into the IP address `35.1.1.48`. All traceroute settings are at their default values. Figure 4-4 is an example of a Tracerout.

```

traceroute to nis.nsf.net (35.1.1.48), 30 hops max, 56 byte packet
 1 helios.ee.lbl.gov (128.3.112.1) 19 ms 19 ms 0 ms
 2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
 3 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 39 ms
 4 ccn-nerif22.Berkeley.EDU (128.32.168.22) 39 ms 39 ms 39 ms
 5 128.32.197.4 (128.32.197.4) 40 ms 59 ms 59 ms
 6 131.119.2.5 (131.119.2.5) 59 ms 59 ms 59 ms
 7 129.140.70.13 (129.140.70.13) 99 ms 99 ms 80 ms
 8 129.140.71.6 (129.140.71.6) 139 ms 239 ms 319 ms
 9 129.140.81.7 (129.140.81.7) 220 ms 199 ms 199 ms
10 nis.nsf.net (35.1.1.48) 239 ms 239 ms 239 ms

```

Figure 4-4. Tracerout example

To start the inquiry, execute:

```
Local> traceroute nis.nsf.net [ENTER]
```

The output is as follows:

The three query packets produce the three roundtrip times recorded in milliseconds and displayed beside the IP addresses.

Traceroute Error Messages

If errors are encountered during execution of the traceroute inquiry, the following codes are returned to the server:

645—Target host not found.

294—No UDP port available.

295—No Traceroute session available. (The server supports up to four simultaneous Traceroute sessions.)

296—Received invalid error message from Traceroute.

Upgrading ENIC Software

You can easily upgrade your ENIC software as follows:

- You can convert a *single*-protocol (LAT-*only* or TCP-*only*) ENIC to a *dual*-protocol (LAT-TCP/IP) ENIC
- You can add TN3270 protocol to an ENIC
- You can add a multiple server download/site feature to a server

You do not need to replace any components in the ENIC to activate these upgrades. Activate the software upgrades by using special password keys.

Activating ENIC Upgrades

To activate any of these new features in your ENIC, contact your technical service representative. Your representative will review the upgrade procedure with you and will provide you with a password key for each ENIC upgrade. When you have this information, proceed as follows:

1. You must be a privileged user to execute the ENIC upgrade commands. Set your port to privileged status. Execute:

```
Local>      set privileged [ENTER]

Password> {privileged password} [ENTER]
```

- 2a. To upgrade from single-protocol to dual-protocol, execute:

```
Local>      define protocol dual enable password
             {password key} [ENTER]
```

- 2b. To add TN3270 support to the ENIC, execute:

```
Local>      define protocol tn3270 enable password
             {password key} [ENTER]
```

- 2c. To enable the ENIC to distribute upgrade features to other servers, execute:

```
Local>      define protocol upload enable password
             {password key} [ENTER]
```

where:

PASSWORD KEY—Specifies an alphanumeric character string to enable the software upgrade on the ENIC. The string is supplied by the manufacturer and is unique to the ENIC.

3. Reset the ENIC. Execute:

```
Local> reset [ENTER]
```

4. Reboot the ENIC with the new features. Execute:

```
Local> crash [ENTER]
```

Removing ENIC Upgrades

You can remove upgrades from an ENIC at any time. For example, you may decide to remove an upgrade feature from an ENIC when that feature no longer is needed on that ENIC. Removing an upgrade may yield extra memory space for other applications, or it may enhance the speed at which the ENIC processes packets.

1. You must be a privileged user to remove ENIC upgrades. Set your port to privileged status. Execute:

```
Local> set privileged [ENTER]
```

```
Password> {privileged password} [ENTER]
```

2. To remove all upgrades from the ENIC and return it to its factory default protocols, execute:

```
Local> define protocol default [ENTER]
```

3. Reset the ENIC. Execute:

```
Local> reset [ENTER]
```

4. Reboot the ENIC. Execute:

```
Local> crash [ENTER]
```

Activating the UNIX Command Set

You can quickly configure the server to accept either DECserver-style or UNIX-style commands. By specifying the type of user interface for a server port, you can instruct the server either to support commands entered in standard DECserver-type syntax or to convert UNIX-type syntax into DECserver-type syntax. Proceed as follows:

1. You must be a privileged user to change the server interface type. Execute:

```
Local>      set privileged [ENTER]
```

```
Password> {privileged password} [ENTER]
```

2. Activate the UNIX command interface on a specified port. Execute:

```
Local>      change port {number of port}  
            user_interface UNIX [ENTER]
```

3. You now are able to execute any UNIX command supported by the server UNIX command set. For example, to obtain a list of port characteristics, execute the UNIX command **stty** as follows:

```
Local>      stty [ENTER]
```

For more information about UNIX commands, refer to appendix D, which is a table of UNIX commands.

Appendix **A**

Glossary of Variables

The following are descriptions of command variables found in several sections of this manual. They are listed in alphabetical order. Not all command variables are listed here.

ACCESS DYNAMIC—Permits a server port to accept local or remote connections.

ACCESS LOCAL—Allows local server users to log into the target port. Remote hosts cannot connect to the port.

ACCESS REMOTE—Permits network users (who are not physically attached to the server) to connect with the target port. In the case of connecting a printer, select Access Remote if the printer does not contain a keyboard, and thus, will never be used as a terminal. Select Access Dynamic if the printer contains a keyboard and may be used, when necessary, as a terminal.

AUI—Used with the Auto-Ethernet selection feature, this variable instructs the server to use the Thick-Wire (DIX/AUI) adapter.

AUTOBAUD DISABLED—Turns off the ability of a port to sense the speed of incoming data characters.

AUTOINIT ENABLE—If Autoinit is not enabled, and a user issues a Stop Interface command and then reboots the server, the interface will not automatically restart when the server reboots. Adding Autoinit Enabled to the command line assures that the interface is automatically restarted when the server is rebooted.

AUTOPROMPT DISABLED—Instructs the port to wait for the user to press the **[ENTER]** key before generating a user prompt. That is, when a user connects to the dial-out modem service, the user must press **ENTER** to see a prompt.

AUTO-SELECT—Senses which of the possible Ethernet adapters is active, and automatically sets the server for that interface. If you do not specify either of the available Ethernet adapters, the server factory setting is Auto-Select. If none of the server adapters is active. For example, no Ethernet cables are attached to the server. The server immediately initializes to permit users to execute configuration commands.

BAUD RATE—Speed at which data characters pass through the port in units of bytes per second.

BREAK DISABLED—Turns off the action of the **BREAK** key at the user's port. If a user presses **BREAK**, nothing will happen.

COMPRESSION—Choices are: enabled, active or disabled. *Enabled* passively defers control to the server at the opposite end of a link. If the other end of the line sends compressed packets, this end will return compressed packets. *Active* immediately forces the line into compression mode. *Disabled* turns off compression regardless of the state of received packets.

DEDICATED NONE—Allows users to connect to the port without automatically connecting to a network service.

DSRLOGOUT—Used with Port Modem Control. DSR logout must be disabled when the Port Modem Control is set to DSR/DTR flow control enabled.

DTRWAIT ENABLED—Instructs the port to assert the DTR and RTS signals only in response to an incoming RING signal or a remote connection to the port.

FLOW CONTROL TYPE—Specifies the method used to manage the movement of data characters through the designated server port. **Note:** Do not use XON/XOFF flow control for SLIP links. SLIP does not recognize these characters.

INACTIVITY LOGOUT—Logs out the server port when the user's terminal becomes idle for a sustained period while at the `Local>` prompt.

MASK 255.255.255.0—In Telnet security, specifies that all digits in the first three bytes of the Telnet IP address are passed to the result, which is compared to the security address.

SECURITY 1—When configuring Telnet security, this identifies the Telnet security structure (in this example, structure #1).

STATE UP—Activates an interface.

10BASET—Causes the server to configure itself for a 10BaseT connector.

THINWIRE—Instructs the server to configure itself for the thin wire adapter.

VCX - Virtual Circuit Exchange, asynchronous Data PBX, Stat Mux, X.25 Pad, FRAD.

VCX GATEWAY - A Gateway between Ethernet, 3270, X.25, Modems and the VCX product line.

WARM—Is a diagnostics feature allowing a user to restart the server, without rebooting; that is, when the command is executed, the server restarts itself using all of the current configuration values.

Default Keyboard Mapping Profiles and Worksheets

This appendix provides a list of key labels found on both the default VT100 (and VT220 for some keys) KMP and the default 3270 KMP. You can photocopy the table to add the custom keystrokes for your KMP file. Table columns contain the following information—

Key #—This is only an indexing number. Use it to reference a particular key.

3270 Key Label—This is the physical label on the 3270 keyboard key.

VT Key—Use this area of the worksheets to record customized keystrokes defined for your custom KMP. The columns headings are:

- **Default Keystroke**—The default control sequence representing this VT keyboard key. This value cannot be changed within the default KMP. Except where noted, these keystrokes represent defaults for VT100, VT220, VT320, VT330, VT340 and VT420. Some keys have two sequences given, one for **VT100** and one for **Other VTs**. The sequence for VT100 is only for the VT100. The sequence for Other VTs is for all other VT terminals *except* the VT100.
- **Custom Keystroke**—If you assign a new control sequence to this key while defining a custom KMP, write the new sequence here. Write the name of your profile at the top of the column. The case of text inside brackets (example: <esc>) is not important.
- **Label**—This is the actual, physical label on the VT keyboard key. A blank here means that a VT key for the indicated 3270 key does not exist.

Keys beyond INDEX #64 cannot be changed and do not appear when you execute a **Show kmp** command.

Key #	3270 key label	VT Key (applies to ALL VT terminals, except where noted)		
		Default keystroke on VT terminal	Custom keystroke (for custom KMP profile)	Label on VT terminal keytop
1	PA1	<ESC>OS		PF4
2	PA2	<ESC>OM		<KP> - (hyphen)
3	PA3	<ESC>OL		<KP> , (comma)
4	CLEAR	^E		
5	ENTER	<F19>		F19
6	PF1	<ESC>1		<KP>ENTER
7	PF2	<ESC>2		
8	PF3	<ESC>3		
9	PF4	<ESC>4		
10	PF5	<ESC>5		
11	PF6	<ESC>6		
12	PF7	<ESC>7		
13	PF8	<ESC>8		
14	PF9	<ESC>9		
15	PF10	<ESC>0		
16	PF11	<ESC> -		
17	PF12	<ESC> =		
18	PF13	<ESC>OP		PF1
19	PF14	<ESC>OQ		PF2
20	PF15	<ESC>OR		PF3
21	PF16	<ESC>Ow		<KP>7
22	PF17	<ESC>Ox		<KP>8
23	PF18	<ESC>Oy		<KP>9
24	PF19	<ESC>Ot		<KP>4
25	PF20	<ESC>Ou		<KP>5

Key #	3270 key label	VT Key (applies to ALL VT terminals, except where noted)		
		Default keystroke on VT terminal	Custom keystroke (for custom KMP profile)	Label on VT terminal keytop
26	PF21	<ESC>Ov		<KP>6
27	PF22	<ESC>Oq		<KP>1
28	PF23	<ESC>Or		<KP>2
29	PF24	<ESC>Os		<KP>3
30	RESV_1	<NULL>		
31	RESV_2	<NULL>		
32	RESV_3	<NULL>		
33	RESV_4	<NULL>		
34	RESV_5	<NULL>		
35	RESV_6	<NULL>		
36	CUR SEL	^Q		
37	RESET	<ESC>Op		<KP>0
38	DISCONN	^B		
		<ESC>[19~		<F8>
39	DUP	<ESC>[20~		<F9>
40	FMARK	<ESC>[21~		<F10>
41	EINP	<ESC>[17~		<F6>
42	EEOF	<ESC>[18~		<F7>
		<ESC>[23~		<F11>
		<ESC>[24~		<F12>
		<ESC>[25~		<F13>
		<ESC>[26~		<F14>
		<ESC>[28~		<F15>

Key #	3270 key label	VT Key (applies to ALL VT terminals, except where noted)		
		Default keystroke on VT terminal	Custom keystroke (for custom KMP profile)	Label on VT terminal keytop
		<ESC>[29~		<F16>
		<ESC>[31~		<F17>
		<ESC>[32~		<F18>
		<ESC>[33~		<F19>
		<ESC>[34~		<F20>
43	DELETE	VT100: <Delete> Other VTs: <ESC>[3~		DELETE
44	INSRT	VT100: <ESC>On Other VTs: <Insert>		<kp>. (period)
45	TAB	^I		TAB
46	BTAB	^H		
47	NL	^M		RETURN
48	HOME	VT100: ^J Other VTs: <FIND>		FIND
49	UP	<ESC>[A~		UP
50	DOWN	<ESC>[B~		DOWN
51	RIGHT	<ESC>[C~		RIGHT
52	LEFT	<ESC>[D~		LEFT
53	LEFT2	^Z		
54	RIGHT2	^X		
55	SPACE	<SPACE>		
56	CENTER	^A		

Key #	3270 key label	VT Key (applies to ALL VT terminals, except where noted)		
		Default keystroke on VT terminal	Custom keystroke (for custom KMP profile)	Label on VT terminal keytop
57	REFRESH	^L		
58	TOGGLE	^N		
59	Reserved	<NULL>		
60	Reserved	<NULL>		
61	Reserved	<NULL>		
62	Reserved	<NULL>		
63	Reserved	<NULL>		
64	Reserved	<NULL>		

Key Mapping

IP-Over-X.25 Communications

This appendix is designed to be used with the *X.25 PAD Card User's Guide*. There are two general sections—

- *IP-over-X.25 Communications*—This explains why IP communications support has been added to the X.25 PAD card and how the PAD card design conforms to current X.25 communications standards. A simple example illustrates how IP packets are transmitted via X.25. A summary of compliance specifications closes the section.
- *Getting Started with IP-over-X.25*—This section outlines the procedure to configure the server to support bidirectional IP-over-X.25 communications. An example illustrates each step.

IP-over-X.25 Communications

IP-over-X.25 packet encapsulation software on the X.25 PAD card expands the X.25 connectivity structure to users of IP. It permits transmission of Internet Protocol (IP) data packets over X.25 Wide Area Networks (WANs). Routing IP datagrams through X.25 WANs enlarges the population of hosts accessible to IP users logged into the X.25 PAD card in a Communications Server. See figure C-1.

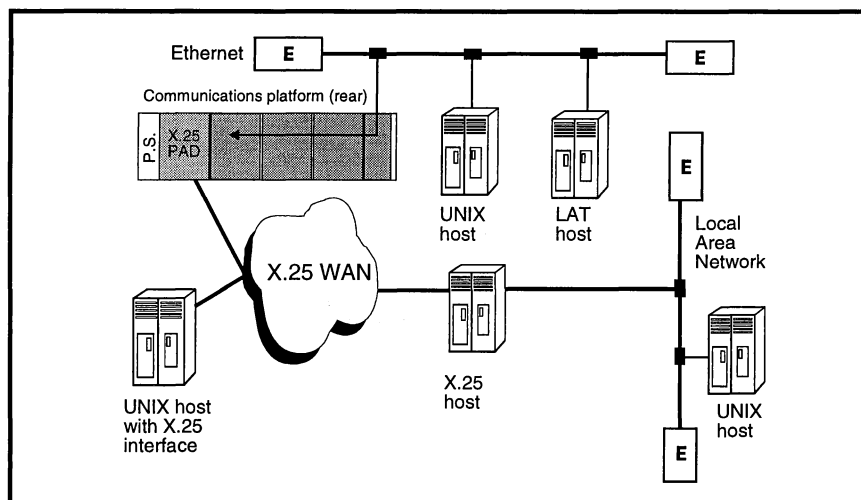


Figure C-1. UNIX users exchange packets via X.25 WAN

Conforming to X.25 Standards

There are several concurrent sources of standards addressing both hardware and software aspects of X.25 communications.

One source, the ITU-T (International Telecommunications Union—Telecommunications Sector, formerly CCITT) periodically publishes recommendations describing how to connect hardware; that is, Data Terminal Equipment (DTE)—typically a computer system or terminal—to Data Communications Equipment (DCE), which is the packet-switched data network providing the functions required to start, maintain and terminate a connection between DTEs.

Other influential groups, such as the International Standards Organization (ISO) and the Defense Data Network (DDN), publish standards to use X.25 protocol to transfer data between computer hosts via the X.25 connections. For example, DDN standards describe two versions of DDN X.25 protocol—

- *DDN Basic X.25 Service*, in which the data portion of each X.25 packet is unstructured. DDN Basic service provides X.25-level source-to-target call management to guarantee delivery of the data packets. Only basic service users can interoperate with other basic service users. DDN hosts that do not implement IP protocol and which use DDN basic X.25 service *cannot* communicate across gateways.
- *DDN Standard X.25 Service*, which provides local-DTE-to-local-DCE support of X.25 connections. In Standard service, X.25 packets carry data in the form of a structured Internet packet, which is the mechanism for IP datagram transmission over X.25 WANs. When DDN Standard service is requested at the time a call is established, the call is established between the DTE and a local X.25 entity. This entity extracts the IP datagrams from the X.25 data packets for transmission through the DDN internet. (See figure C-2.)

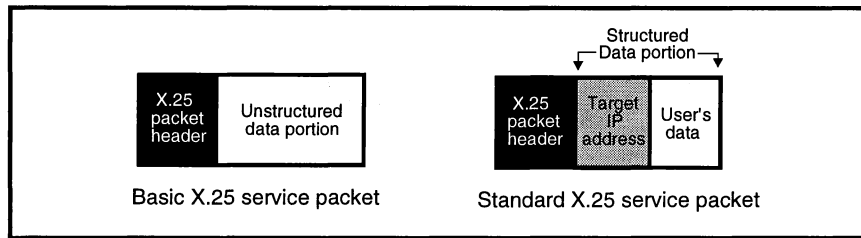


Figure C-2. Supports basic and standard service

The Communications Server's X.25 PAD card supports basic and standard service.

In response to the demands of TCP users for X.25 connectivity to both public and private networks *and* in the absence of a ITU-T standard (the ITU-T currently has not published a recommendation describing standards for encapsulating IP datagrams within X.25 packets), developers at commercial and educational institutions have devised their own IP encapsulation strategies. Despite the experimental nature of these strategies, the more successful solutions have been documented as RFCs and have been widely copied. It seems likely that the ITU-T will eventually adopt the best aspects of the most workable IP encapsulation strategies.

The new X.25 PAD card couples the features of the most popular IP encapsulation strategies with current Defense Data Network (DDN) X.25 service standards.

Theory of Operation

IP-over-X.25 operations are best illustrated through the following example:

- Consider a user, logged into dual-protocol (LAT/TCP) *Server1* on *LAN1*, who wants to connect to a service called *ACCOUNT*, which resides on a UNIX host on *LAN2*.
- What makes this connection possible is that *LAN1* is linked to *LAN2* via an X.25 WAN. See figure C-3. *Server2* and *Server3* (each equipped with a dual-protocol ENIC) connect their respective networks to the WAN via X.25 PAD cards featuring IP-over-X.25.

The connections steps are illustrated in figure C-3.

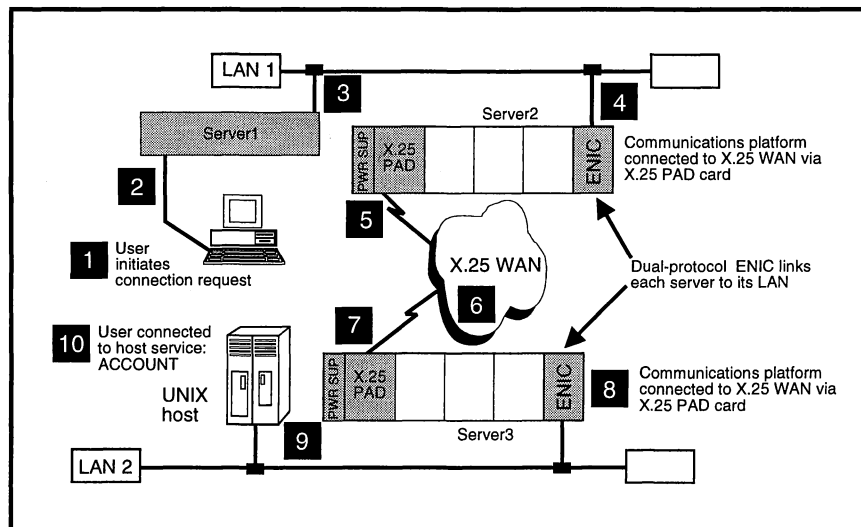


Figure C-3. Connecting for IP-over-X.25 operation

Starting the Connection

The following ten steps trace the path of packets from the source of the connection request (user's terminal) to the service residing at the target host. The X.25 PAD card already is configured for IP-over-X.25 connections.

1. A user (logged into *Server1* on *LAN1*), wanting to start a connection to a target IP host *ACCOUNT* on *LAN2*, executes:

Local> connect account [ENTER]
2. *Server1* accepts the user's connection request and quickly determines that the target host is not on the local network.

3. Unable to complete the outbound connection without assistance, *Server1* consults its routing table and sees that *Server2* knows a route to the target host, so it transmits the IP packet through the LAN to *Server2*.
4. The ENIC in *Server2* accepts the IP packet and looks in its data base for the target IP address and its corresponding X.121 address. When it finds the relationship, it sends a request to the X.25 PAD card (residing in one of its slots) to open up one of the unused, allocated PAD ports to send the data to the target IP host.
5. The X.25 PAD card checks for a link through the X.25 WAN to the target host.
 - If there is no link; that is, no virtual circuit exists between the X.25 PAD card on *Server2* and its counterpart on *Server3*, the X.25 PAD assigns a logical channel number (LCN) to the circuit and dispatches an X.25 call request packet containing the LCN and the target X.121 address. When the remote DTE accepts the call and a virtual circuit is established, data packets can be transferred over the new link between the source and target PAD ports.
 - If there is a link between the X.25 PAD card on *Server2* and its counterpart on *Server3*, the X.25 PAD card encapsulates IP data packets into an X.25-compatible format and dispatches them to the target host.
 - If the link is *down* and cannot be restarted, the X.25 PAD card returns an error message indicating that the connection cannot be completed.

IP data packet encapsulation involves adding both a string of prefix bits (called a *header*), containing the LCN, to the front end of the packet, and a string of suffix bits (called a *trailer*) containing packet sequence numbers, to the end of the packet. The IP datagram travels within the frame created by the encapsulation. When a single IP data packet is too large to traverse the X.25 WAN in its encapsulated form, it can be broken into as many as 14 separate packets. To maintain continuity, the *More Data* bit is set in each of the separate packets (except for the last in the series.)

6. The X.25 PAD card in *Server2* dispatches packets (through the X.25 WAN) to the X.25 PAD card residing in *Server3* (on *LAN2*).
7. The target X.25 PAD card in *Server3* receives each data packet and removes the LCN from the frame. The remaining portion of the frame—the IP datagram—contains both the target IP address of the Telnet service *ACCOUNT*, configured on the UNIX host residing on *LAN2* and the user's data.
8. The ENIC in *Server3* accepts the IP packet from the X.25 PAD card.

9. The ENIC in *Server3* forwards the IP packet along *LAN2* to the destination IP address.
10. The *LAN1* user's data packets arrive at the UNIX host and the target Telnet service *ACCOUNT*. The same process is reversed when an IP packet is transmitted from a source IP address on *LAN2* to the user's target IP address on *LAN1*.

Connecting from a LAT-Only Source

To connect from a LAT-only server to an IP host that is only accessible through an X.25 WAN, use a Network Protocol Translator (NPT) device (such as an NPT card in a Communications Server) to convert the LAT protocol packets to TCP/IP packets. The resulting IP packets can be encapsulated and transmitted to the target host.

Ending the Connection

Passive Disconnection: A server timer monitors the level of datagram traffic through the server port link. When no activity is detected across the link for a finite period, the link is dropped.

Active Disconnection: Log out of the port supporting the IP/X.25 link. Otherwise, wait for the timeout.

Specifications

Following are the specifications with which this implementation of IP-over-X.25 complies—

- **DDN Specifications**—100% compliant with Classes A, B and C
- **IP Specifications**—Fully compatible with MIL-STD-1777
 - IP complies with Request for Comments (RFC) numbers 791, 792 (ICMP), 1122, 1236 (DDN mapping)
 - 100% compliant with 1236 (CCITT 1984)
- **X.25 Specifications**—100% compliant with FIPS 100/Federal Standard 1041

For a complete listing of all other specifications with which the Communications Server is compliant, see the *Communications Server Installation Guide*.

Getting Started with IP-Over-X.25

A procedure follows to configure an X.25 PAD card-equipped Communications Server for IP-over-X.25 data communications support. Server and PAD configuration parameters are set at both the local and remote servers. When the configurations are complete, local server users can connect (transparently through the local PAD card) to target IP hosts via the X.25 WAN.

Similarly, IP users logged into IP hosts on the remote Ethernet can connect through a similarly configured remote Communications Server via the X.25 WAN to IP hosts on the local network. Information in figure C-4 shows how to configure Communications Server 1 so that IP user at Ethernet 1 can connect (via X.25 WAN) to IP host on Ethernet 2. The X.25 PAD card in Server 1 encapsulates IP packets and dispatches them to Server 2. There they are returned to their IP format for delivery to the target host.

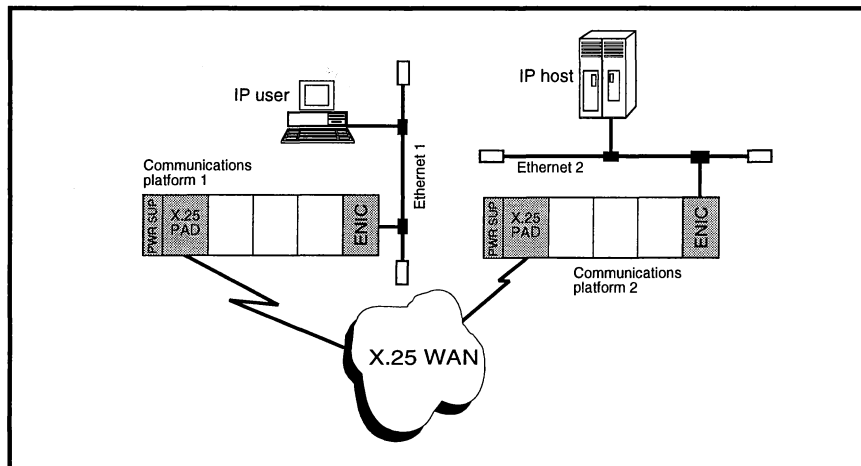


Figure C-4. Configuring for remote Ethernet connections

The major parts of the procedure are as follows—

- Configuring the local PAD card hardware
- Configuring local server ports
- Configuring the local PAD
- Adding addresses to the local X.121 data base
- Configuring routes to destinations
- Repeating all configuration procedures at the remote server

Configuring the Local PAD Card Hardware

The X.25 PAD card contains four removable clocking jumpers that must be properly installed to support the specified source of software clocking. Internal clocking means the clock control comes from inside the X.25 PAD card. External clocking means the clock control comes from a device outside the PAD card, such as a modem.

To determine the type of clocking control required in your configuration, first consider how the X.25 PAD card will be connected. Figure C-5 illustrates two hardware configurations—

- *Back-to-back*, in which the X.25 PAD cards are connected via a dedicated cable
- *Modem-to-modem*, in which each PAD card is connected to a modem, which, in turn, is connected to the X.25 WAN

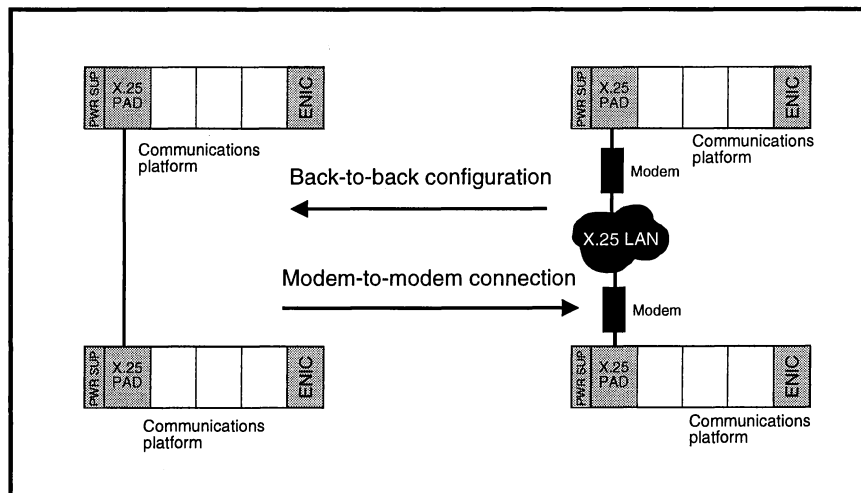


Figure C-5. Hardware configurations for connecting the PAD cards

Table C-1 lists hardware configurations (illustrated in figure C-5) and indicates the appropriate clock control settings at PAD 1 and PAD 2. For example, in a back-to-back configuration in which PAD 1 controls the clocking internally on the link, PAD 2 must be set to *external* clocking. Observe that in the typical modem-to-modem configuration—in which each local modem supplies the PAD clock control—both PAD cards must be set for *external* clocking.

1. To use the table, locate the appropriate hardware configuration in the first column. Then, read across that row to the settings for each of the PAD cards. Set clocking on each PAD card as shown in the table. Use the link mode settings to configure the PAD software.

Table C-1. Clock Control Settings

Hardware configuration	Set clocking at PAD 1 to:	Set clocking at PAD 2 to:
Back-to-back	Internal/mode DCE	External/mode DTE
Back-to-back	External/mode DTE	Internal/mode DCE
Modem-to-modem	External/mode DTE	External/mode DTE

2. Select the type of PAD interface path. A toggle switch—located on the PAD card backplate—flips left (V.11/X.21) or right (V.35) to set the desired path.

CAUTION – The X.25 PAD card contains static-sensitive devices. You must be properly grounded before touching the PAD card, its firmware devices and its interface jumpers.

For complete instructions to locate and set the clocking and interface jumpers, refer to Section 2 of the *X.25 PAD card User's Guide*.

Configuring Local Server Ports Summary

The steps involved in configuring local server ports are summarized below. A detailed description of each step is given in the following pages.

1. Log onto the Communications Server.
2. Set the port you'll be using to privileged status.
3. Display the configuration of the local Communications Server.
4. Determine the X.25 card's slot number, and the numbers of the server ports assigned to the X.25.
5. Configure the server ports that are assigned to the X.25 card.
6. Create a local PAD server.
7. Connect to the new local PAD server.
8. Wait to see the X.25 salutation and local PAD prompt.
9. Switch your PAD port to privileged mode.
10. Assign an X.121 address to the local PAD.
11. Choose a network interface type to be supported by the local PAD (Telenet, NET2, TYMNET, DDN).
12. Assign a clocking source setting to the PAD software.
13. Log out of the local PAD.
14. Create an IP/X.25 access service and assign the dedicated PAD ports to this service.
15. Populate the local X.121 database with the target X.121 address of each remote PAD with which the local PAD will connect
16. Confirm the IP address assignments in the data base.
17. Create a "dummy" dynamic interface to the server.
18. Confirm the interface entry in the server data base.
19. Create a route to each of the destinations of the X.25 link.
20. Configure the server to transmit packets through the local PAD.
21. Repeat the entire procedure for the target server.

Configuring Local Ports

1. Log into the local Communications Server that will be configured to support IP-over-X.25 communications.

```
Enter username> {username string} [ENTER]
Local>
```

2. You will be using privileged commands. Therefore, set the port through which you are operating to privileged status. Execute:

```
Local>      set privileged [ENTER]
Password> {privileged password} [ENTER]
```

where:

PRIVILEGED PASSWORD—Is the string required to activate privileged port operations. The server factory default is `system`, however, this password may have been changed. Consult the server manager for further information.

3. Display the current configuration of the local Communications Server (the Executing Version columns only display if the server is downloading images). Execute:

```
Local> show configuration [ENTER]
```

4. Using the Show Configuration display, determine both the number of the slot containing the X.25 PAD card and the numbers of the server ports assigned to the PAD card. For example, figure C-6 shows the X.25 PAD card residing in server slot number two, and using ports 33 to 64.
5. Configure the server ports assigned to the X.25 PAD card. Execute:

```
Local>      change port {numbers of ports} access
            dynamic autobaud disable virtual enable flow
            disable absolute enable [ENTER]
```

where:

{NUMBERS OF PORTS}—Specifies the numbers of the server ports assigned to the X.25 PAD card.

ACCESS DYNAMIC—Permits the server ports to both initiate outgoing calls and receive incoming calls.

AUTOBAUD DISABLE—It is recommended that the **Autobaud** feature be disabled at ports set for dynamic access.

ROMed Version # of ROMed Images = 10				Executing Version: # of Executing Images = 11			
#1	-	(C083)	Rev. P	#1	-	(C083)	Rev. P
#2	-	(C1A2)	Rev. F	#2	-	(035E)	Rev. G
#3	-	(C258)	Rev. G	#3	-	(C258)	Rev. G
#4	-	(C339)	Rev. L	#4	-	(0232)	Rev. K
#5	-	(C465)	Rev. Q	#5	-	(C465)	Rev. Q
#6	-	(C551)	Rev. L	#6	-	(C551)	Rev. L
#7	-	(C608)	Rev. E	#7	-	(13AA)	Rev. B
#8	-	(C63D)	Rev. C	#8	-	(1036)	Sli. 3
#9	-	(C643)	Rev. F	#9	-	(C643)	Rev. F
#10	-	(CDB6)	Rev. S	#10	-	(14EF)	edit 023t
				#11	-	(040F)	P3B
FEP Space Available: 128kb							
Backplane Configuration -							
Slot #1 -	Rev. 1.00	[VCP/LC-RS423-32]		Ports: 1-32			
Slot #2 -	Rev. 1.00	[VCP/LC-X.25]		Ports: 33-64			
Slot #3 -	Rev. 1.00	[VCP/LC-RS423-16]		Ports: 65-80			
Slot #4 -	Rev. 1.00	[VCP/LC-RS232-8]		Ports: 81-88			

Figure C-6. Server configuration w/X.25 PAD card

VIRTUAL ENABLE—Activates the virtual login feature on the specified ports. When an incoming call arrives at the X.25 PAD card the session begins and the port becomes active, just as though the user pressed **ENTER ENTER**.

FLOW DISABLE—Deactivates data flow control at the specified ports.

ABSOLUTE ENABLE—Activates the Server Absolute Timer logout feature for the specified ports. Inactive ports are logged out from all active sessions through the server upon expiration of the server absolute time limit. This **ONLY** occurs if there is **NO** data activity on the port.

Configuring the Local PAD

6. Create a local PAD management service. Use this service to connect to the local PAD to configure the PAD parameters required for IP encapsulation. Execute:

```
Local> change service {name of PAD management service} port
      {server port number} virtual enable virtual pad
      [ENTER]
```

where:

{NAME OF PAD MANAGEMENT SERVICE}—Select a string of up to 16 alphanumeric characters describing the locally-offered service that will provide access to the PAD.

{SERVER PORT NUMBER}—Specifies the number of the server port that is dedicated to the PAD management service. It is good practice to assign the highest numbered port to this service. For example, in figure C-6, the highest numbered port is 64. In that case, the PAD management service is assigned to port 64.

VIRTUAL ENABLE—Permits an automatic connection through the designated port to a target designated by the string of virtual text, which follows.

VIRTUAL PAD—Indicates the target of the virtual connection. When a local server user connects to the specified local service, the virtual feature automatically routes the connection to the local X.25 PAD.

For example, to create the local PAD management service `x25mgr` and assign it to port 64 of the local server, execute:

```
Local> change service x25mgr port 64 virtual enable
      virtual pad [ENTER]
```

If the local PAD management service will be accessible to IP users, execute:

```
Local>  change service {name of PAD management service} port
        {server port number} virtual enable virtual pad
        telnet enable tcp port 23 ip {IP address of local
        server} [ENTER]
```

7. Connect to the newly-created PAD service. Execute:

```
Local>  connect {name of PAD management service} [ENTER]
```

For example, to connect to the service `x25mgr`, execute:

```
Local>  connect x25mgr [ENTER]
```

8. Upon accepting the connection request, the server automatically completes the virtual connection to the local PAD. Upon completing the connection to the local PAD, the server displays the X.25 salutation and the local PAD> prompt.

```
Welcome to the X.25 PAD card
```

```
PAD>
```

9. To execute local PAD configuration procedures, your PAD port must be operating in privileged mode. To switch your PAD port to privileged mode, execute:

```
PAD>      set privileged [ENTER]
```

```
PASSWORD> {privileged password string} [ENTER]
```

where:

{PRIVILEGED PASSWORD STRING}—Is the string of alphanumeric characters required for entry to privileged mode. The PAD factory default is **system**. However, the default password may have been changed. Consult your system manager for details.

When your connection has been switched into privileged mode, the screen displays the following:

```
Privileged Mode
```

```
PAD>
```

10. Assign an X.121 address to the local PAD. Execute:

```
PAD>      change x25 address {source X.121 address} [ENTER]
```

11. Select the type of network interface that will be supported by the local PAD. The options are Telenet, DDN, NET2, and Tymnet.

To change the network type, execute:

```
PAD>      change network {network type} [ENTER]
```

12. Assign a clocking source setting to the PAD software. This is the same setting you used to configure the local PAD card hardware.

```
PAD>      change clock source {clocking source} link mode {mode type} [ENTER]
```

where:

{CLOCKING SOURCE}—Indicates either internal or external, depending upon the hardware configuration. See table C-1.

{MODE TYPE}—Indicates either DTE or DCE, depending upon the clocking source. See table C-1.

For example, in a back-to-back configuration, where PAD1 is dedicated to PAD2, configure the PADs as follows:

```
PAD1>      change clock source internal link mode dce [ENTER]
```

```
PAD2>      change clock source external link mode dte [ENTER]
```

13. The local PAD configuration is complete. Log out of the local PAD. Execute:

```
PAD>      logout [ENTER]
```

Creating an IP/X.25 Access Service

14. You are returned to the local server. Create an IP/X.25 access service and assign the dedicated PAD ports to this service. The service name associates the ports to target IP and X.121 address entries in the X.121 data base. Execute:

```
Local>      change service {name of local IP/X.25 access service}
port {pool of server port numbers} pool enable lat
disable telnet disable [ENTER]
```

where:

{NAME OF LOCAL IP/X.25 ACCESS SERVICE}—Indicates the name of the local service that the ENIC uses to access the pool of server ports assigned to the X.25 PAD card. For example, if X.25 PAD cards are installed in server slots 1 and 2, ports one to 32 are assigned to the PAD 1 pool; and ports 33 to 64 are assigned to the PAD 2 pool. A unique service name is assigned to each PAD pool.

{POOL OF SERVER PORT NUMBERS}—Indicates the local server port numbers assigned to the X.25 PAD card. It is good practice to assign the highest PAD port number to a PAD management service (see *Step 6*). The remaining pool of ports is assigned to the PAD access service. For example, if a PAD card resides

in slot one, ports one to 31 are assigned to the pool, while port 32 is reserved for the PAD management service.

POOL ENABLE—Instructs the ENIC to regard the specified server ports as a pool of ports. See the **Change/Define/Set Service Pool** command.

LAT DISABLE—Prevents LAT users from connecting to the pool of server ports.

TELNET DISABLE—Prevents Telnet users from connecting to the pool of server ports.

Adding Addresses to the Local X.121 Data Base

15. Add addresses to the local X.121 data base with the target X.121 address of each remote PAD with which the local PAD will connect. The command syntax depends upon the type of network interface that will be supported by the local PAD. The two options are Telenet or DDN.

If the interface is Telenet, execute:

```
Local>  change x121 x121_addr {X.121 address string of remote
        PAD} ip_addr {IP address of remote PAD} service {name
        of local PAD access service} [ENTER]
```

If the interface is *DDN*, execute:

```
Local>  change x121 ip_addr {IP address of remote PAD} ddn
        service {name of local PAD access service} [ENTER]
```

where:

{X.121 ADDRESS STRING OF REMOTE PAD}—Specifies the target X.121 address of the PAD at the remote end of the connection.

{IP ADDRESS OF REMOTE PAD}—Indicates the IP address of the remote X.25 PAD. In operation, this IP address serves as a gateway to the destination host. It is good practice to assign a unique IP address to the PAD.

{NAME OF LOCAL PAD ACCESS SERVICE}—See step 14.

16. Confirm the IP address assignments in the server data base. Execute:

```
Local>  show address [ENTER]
```

The following screen is displayed:

Internet Address	IFC#
{server's primary IP address}	0
{PAD's unique IP address}	0

Configuring Routes to Destinations

17. Create a “dummy” dynamic interface to the server. The dummy interface is used as a placeholder to set up routes through the X.25 link to the target host. Execute:

```
Local> change interface 63 [ENTER]
```

18. Confirm the interface entry in the server data base. Execute:

```
Local> show interface [ENTER]
```

You will see the screen shown below.

IFC #	Type	Port #	Mtu	Internet Address	State
0	Ethernet	N/A	1500	Default	Bdcast, Init, UP, Valid
63	Dynamic	N/A	1620	Default	Init, UP, Valid

IP Over X.25

19. Create a route to each of the destinations that will be accessed via the X.25 link. Execute:

```
Local> change route {route number} destination {IP address of
target host} gateway {IP address of target PAD}
interface 63 type network [ENTER]
```

where:

{ROUTE NUMBER}—Select an unused server route number and assign it to this route. To determine available routes, execute a show routes command.

{IP ADDRESS OF TARGET HOST}—Specifies the IP address of the destination IP host.

{IP ADDRESS OF TARGET PAD}—Specifies the IP address of the remote PAD through which the connection to the remote host is made.

INTERFACE 63—Instructs the server to use the “dummy” interface to set up the route to the target host.

TYPE NETWORK—Instructs the target PAD (acting as a gateway) to read the network portion of the destination IP address when routing packets.

For example, to set up route 5 to a host at IP address 12.34.56.78 through a gateway PAD at IP address 12.34.91.23, and to allow the target PAD to read the network portion of the destination IP address when routing packets, execute:

```
Local> change route 5 destination 12.34.56.78 gateway
12.34.91.23 type network interface 63 [ENTER]
```

20. Configure the server to transmit packets through the local PAD acting as an intermediary device between the local server and the remote PAD.

Execute:

```
Local> change ip forward enable [ENTER]
```

21. Repeat entire procedure at target server and target PAD.

Repeating the procedure is necessary only if IP links are to be initiated from the "other end." This of course also assumes that the other end is a server; it could be, as an example, a workstation with an X.25 interface.

Appendix D

UNIX/DECserver Equivalents

Table D-1. Unix Commands and DECserver-Style Equivalents

UNIX command syntax	Equivalent DECserver-style syntax
ARP -a	SHOW ARP
ARP -d {IP address}	CLEAR ARP {IP address} PURGE ARP {IP address}
ARP -s {IP addr} {Eth addr}	CHANGE ARP {IP address} {Ethernet add}
FG {session number}	RESUME {session number}
HOSTS	SHOW DOMAIN
HOSTS -f	CLEAR DOMAIN LOCAL PURGE DOMAIN LOCAL
JOBS	SHOW SESSIONS
KILL {session number}	DISCONNECT {session number}
MAN {topic} {subopic}	HELP {topic} {subtopic}
OPEN {hostname}	CONNECT TELNET {hostname}
RLOGIN {hostname} -1 {userID}	CONNECT RLOGIN {hostname} USER {userID}
STTY STTY-a STTY-g	SHOW PORT
SU	SET PRIVILEGED
TELNET {hostname}	CONNECT TELNET {hostname}
WALL "{message text}"	BROADCAST ALL {message text}
WHO	SHOW USERS

UNIX / DECserver

Error Messages

This appendix contains error messages for the latest ENIC software.

If an error message is generated during server operation, the condition is reported to the user. You will see a unique code number that identifies each message displayed beside a brief explanation of the error message. This facilitates logging incoming messages and improves recognition of a recurring message.

The server factory setting activates the message code feature for all ports. The message numbers and their respective categories applicable to this appendix are as follows—

- **600 - 699** — Server-specific warnings of potential error conditions
- **700 - 799** — User entry execution errors
- **1600 - 1699** — Macro error messages. These messages are directed at the writer of the macro code.

The following table lists message number and associated message text for each message. A description of the condition reported by the message is included. When more than one cause is indicated, the most likely cause is listed first, followed by less likely causes. The appropriate corrective action is listed for each message.

Error Messages

Table E-1. Error Messages, Descriptions, Causes and Actions.

Message	Description	Cause	Action
661 —KMP {KMP name} Not Known	The user specified a Keyboard Mapping Profile not understood by the server.	The specified Keyboard Mapping Profile does not exist and the command cannot be executed.	Execute the Show KMP command without including the KMP name. This will display all Keyboard Mapping Profiles in memory. Select the appropriate profile name and re-enter the command.
662 —Attempt to Modify Default KMP	Attempted to modify control in default KMP.	The server prevents modifications to the default KMP.	Copy the KMP to a new name and execute changes in the new file.
663 —Attempt to Map Unmappable Key Error	Attempt to assign a function to a key in the KMP.	The target key is not an assignable key.	Select another key for this function assignment.
701 — Command Syntax Error	A command was entered improperly (either the command keywords were omitted or the command modifiers were not entered according to the proper format).	A port user has entered an unknown keyword, improperly spelled a keyword or entered the correct keywords in the wrong sequence.	Enter the command again using the proper syntax.
702 — Keyword Not Known or Ambiguous	A command was entered improperly. The keyword in the error message was abbreviated without enough letters to distinguish it from other commands, was misspelled or improperly positioned on the command line).	A port user has entered an unknown keyword, entered keywords in improper order or has not entered the keywords in the proper sequence.	Enter the command again using the proper keyword.
703 —Value Invalid or out of Range, {number}	An improper command.- A numerical value that cannot be adhered to.	User has entered an out of range value when issuing a set, change, or command with a numerical value.	Enter the command again using an acceptable value.

Error Messages

Message	Description	Cause	Action
719— Insufficient Resources to Complete	Attempt to execute a command through the server. Server cannot complete the command because of inadequate memory or that the maximum number of queued requests has been reached.	The local services have reached their maximum limit, and the user has issued a command. The local services cannot support the requested operation.	Execute the Show Server Status command and examine the output. If any of the active nodes, sessions, circuits or other measures of connection activity are at their maximum value, no additional connections currently are possible. If none of the values are at their limit, contact the node system manager for further instructions.
777— Destination Profile {KMP name} Already exists	You attempted to create a new KMP using an existing name.	Only one name is permitted per KMP.	Select another KMP name for the new KMP.
791— Profile Not Known	Attempt to execute either a Show or Load command on nonexistent profile.	User attempted to execute either a Show Profile {name of port} or Load Profile {name of profile} command. The profile name does not exist.	Execute a Show Profile command without including a profile name. This command displays all of the existent profile names. Choose the appropriate profile name.
1639— Syntax Error	The parser encountered an error in macro code syntax.	Generally a syntax error and appears with the line containing offending code.	Correct the syntax error or errors as required.
1640— Failed to Open Source File	The parser read an instruction containing a target file name but was unable to open that file.	acro failed to open a target field specified within the macro code.	Recheck target file name. Maybe you gave it the wrong name; or maybe you don't have permission to read it; or maybe you don't have TFTP set up correctly.

Error Messages

Error Messages

Message	Description	Cause	Action
1641— Unexpected End of File	The parser read the macro file but the file doesn't seem to be complete.	Either the file is truncated or it doesn't have the .. at the end of the file.	Review the macro file for completeness. Reload the complete macro file and retry the macro function.
1642—Invalid Number	The parser read a number that it could not understand.	The number may contain both numerals and characters, or it may be too big (exceeds the allowable value), or number may be garbled.	Recheck all numerical values in the macro file. Correct values as necessary.
1643—Too Many Digits	Parser read a number with too many digits.	Value specified exceeds the maximum allowable digits.	Recheck all numerical values in the macro file. Correct.
1644— Integer Literal out of Range	Parser read an integer that is too big doing the ASCII-to-binary conversion.	The largest allowable integer is 4,000,000,000.	Recheck all integers and reduce any that exceed the limit.
1645—Invalid Expression	The parser read an expression that it could not execute.	You may be allowed to execute simple expressions like "A + B" but you may be trying to execute "A + B + C."	Recheck macro file for illegal logical statements and algebraic expressions. Correct as required.
1646—Invalid Assignment Statement	The parser could not execute an assignment statement.	Statement attempted to either assign a numerical value to a string variable or a string to a numeric variable.	Recheck macro file for illegal assignment statements. Correct as required.
1647— Undefined Identifier	Parser encountered an identifier not previously declared within the macro code.	Using a name not defined or used before; for example, on the right-hand side of an equation.	Recheck all identifiers used in the macro and check that all names are defined.
1648—Invalid Statement	Parser detected a keyword that it could not understand.	Wrong keyword. For example, parser was expecting a WHILE statement, but read a FOR statement.	Recheck all statements in the macro for logical consistency and correct as necessary.

Error Messages

Message	Description	Cause	Action
1649—Unexpected Token	Parser encountered an unexpected token, such as a symbol or a character, not in the code.	A token within the macro codes that appears to have no function. A single symbol such as a plus (+) sign, a minus (-) sign, or a curly brace ({}). a single character or a word (WHILE) or a variable name.	Recheck all statements in the macro for logical consistency and correct as necessary. (Note: It is unlikely this error will occur.)
1650—Invalid for Control Variable			Unlikely this error will occur. <i>FOR</i> control is not supported.
1651—Invalid Constant	Parser encountered a constant value that it could not understand.	May have entered a constant; a bad string, or a number.	Recheck macro code for offending constant. Correct as required.
1652—Redefined Identifier	Parser encountered an identifier that did not match the <i>identifier definition</i> previously established in the macro.	An identifier defined as an integer variable and used it as a string or vice-versa.	Recheck all statements in macro code to verify identifier compatibility.
1653—Invalid Type Specifier	The parser encountered unexpected characters.	Parser expected "INT," a STRING, the dollar sign (\$) or pound sign (#),	Recheck all statements in the macro for logical consistency and correct as necessary.
1654—Invalid Identifier	Parser encountered an identifier that doesn't follow the main conventions.	You attempted to define an identifier that exceeded macro convention limits: max 70 characters, and the leading character must be an alphacharacter.	Recheck macro code. Redefine identifier to conform to macro code convention.
1655—Not An Identifier	Parser encountered an identifier not previously defined within the macro.	Trying to use a name not defined or used before.	Recheck macro code. Verify that the specified identifier is defined, and its usage is consistent.

Error Messages

Error Messages

Message	Description	Cause	Action
1656— Incompatible Types	Parser encountered an integer when it was expecting a string; or vice-versa.	May be trying to add a string to an integer or vice-versa. You only can add an integer to an integer or a string to a string.	Recheck macro and verify that identifiers are used in a manner consistent with their definitions.
1657— Invalid Identifier Usage	Parser encountered an identifier string that could not be validated.	Trying to use an identifier in the code where not allowed.	Recheck macro code for offending identifier.
1658— Incompatible Assignment	Parser detected a string or an integer that could not be validated.	Trying to assign an integer to a string or a string to an integer.	Recheck macro code for offending assignment.
1659— Min Limit Greater Than Max Limit	This problem is related to the FOR control variable.	Unlikely to occur.) The FOR control variable is not sup(ported).	FOR control variable not sup(ported).
1660— Nesting Too Deep (This check may not be implemented.)	Parser has detected a loop that exceeded the allowable number of sublevels.	Too many sublevels; or series of braces; or WHILEs. Up to four levels of WHILEs are allowed.	Review macro code. Adjust code to function with fewer sublevels.
1661— Wrong Number of Parameters	Parser encountered a subroutine in which variable parameters were specified but the numerical values could not be inserted, because of a mismatch.	A subroutine with the wrong number of parameters is being called.	Recheck macro to verify that the number of variable parameters passed to the subroutine matches the <i>number of variable parameters</i> expected by the subroutine.
1662— Invalid Variable Parameter	Parser encountered a subroutine with variable parameters passed to the subroutine, but did not match the type of variable parameter expected.	An integer has tried to pass where the parser is expecting a string or vice-versa.	Recheck macro. Verify that the variable parameters passed to the subroutine match the variable parameters within the subroutine.
1663— Code Buffer Overflow	Code too complex.		Simplify the macro code or shorten the program.

Error Messages

Message	Description	Cause	Action
1664— IO Code Segment Overflow			Attempted to read-in a binary image that was too big for server memory. Server does not support binary images, so it is unlikely this message will appear.
1665— Maximum Identifier Length Exceeded	The parser encountered an identifier whose length exceeds macro code conventions.	An identifier that is too long and has too many characters was created.	Recheck macro to verify that all identifiers comply with macro conventions.
1666— Maximum String Length Exceeded	Parser encountered a string whose length exceeds macro code conventions.	Exceeded the maximum of 255 characters allowed per string.	Recheck macro to verify that string complies with macro conventions.
1667— Unimplemented Feature	Parser encountered a command in the code that does not have an assigned function.	You are attempting to execute a command that is not supported in the current version of the server software.	Refer to Section five, <i>Managing the Stand-alone Servers</i> for a complete list of macro facility features.
1668— Exceeded Number of Arguments Allowed	Parser detected an argument that exceeded macro convention limits.	You have exceeded the five argument limit in a subroutine.	Review macro and adjust code to function with fewer arguments.
1669— Exceeded Number of Declarations Allowed	Parser detected a subroutine exceeding the allowable declarations.	The limit of five declarations has been exceeded.	Review macro and adjust code to function with fewer declarations
1670— Memory Error	No memory in the server.	There are too many operations in server memory at this time.	Reduce size of code.
1671— Memory Allocation Error	Server memory error.	Server could not find space in memory for the macro.	It is unlikely this message will appear.
1672— Memory Free Error	Server memory error.	Server could not purge data from its memory.	It is unlikely this message will appear.

Error Messages

Error Messages

Message	Description	Cause	Action
1673—Read Error	The server was unable to open the specified file from the TFTP host.	The server could not read the file when reading a file from the TFTP host.	Verify that both the path to the TFTP host, and filename are correct.
1675—Symbol Table Access Error	Parser could not execute the macro.	There is an internal error in the software supporting the macro function.	Contact technical support. (May not be user-correctable.)
1676—Not A Server Token	Server was unable to execute a macro.	Parser went to a line in the macro code and stopped.	Examine the offending line of code. The problem may be an internal software problem.
1677—Invalid Variable Type	Parser detected a variable parameters whose usage is not consistent with its definition.	A string or an integer is being used where a variable should be used.	Recheck macro and verify that variable parameters are used consistently with their variable types.
1678—Duplicate Labels	Parser detected a previously used GOTO label.	A GOTO has labels. The same label cannot be used twice.	Recheck macro to verify that GOTO labels are only used once.
1680—Missing Right Parenthesis	Parser detected a left parenthesis and no right parenthesis.	Right parenthesis omitted.	Recheck macro to verify both left <i>and</i> right parentheses.
1681—Missing Left Parenthesis	Parser detected a right parenthesis and no left parenthesis.	Left parenthesis omitted.	Recheck macro to verify both left <i>and</i> right parentheses.
1682—Missing Identifier	Parser encountered an incomplete statement.	A statement containing an incomplete expression.	Check statements in the Macro for logical consistency.
1683—Missing ;	Parser detected a statement without a semicolon (;).	No semicolon was found after the statement.	Recheck macro to verify that each statement is punctuated with a semicolon (;).
1684—Missing Then	Parser read an IF statement without a matching THEN statement.	Every IF statement must have a matching THEN statement.	Locate the line in the code and add the THEN statement.

Error Messages

Message	Description	Cause	Action
1685— Missing Constant	Parser encountered an incomplete statement.	An incomplete expression. For example, having a 1 + (Blank) expression.	Recheck all statements in the macro for logical consistency.
1686— MISSING :	Parser detected a statement without a colon (:).	No colon was found after the statement.	It is unlikely this message will appear.
1687— Missing End	Parser did not detect an END statement.	END statement omitted from the macro code.	An END statement exits the macro. Recheck the macro.
1692— Missing Comma	Syntax error in macro.	Comma missing from macro.	Add comma to macro.
1693— Missing Left Curly Brace	Parser detected a right curly brace (}) without left curly brace ({).	Left curly brace omitted.	Recheck macro and verify that both left and right curly braces are used.
1694— Missing Right Curly Brace	Parser detected a left curly brace ({) without right curly brace (}).	Right curly brace omitted.	Check macro and verify that both left and right curly braces are used.
1696— Macro Is Already Loaded	Server was unable to load the specified macro.	You tried to load a macro into the server previously loaded.	Refer to chapter 4, <i>Managing the Communications Server</i> , for the procedure to remove macros.
1697— There Is No Macro Space Remaining	Server was unable to load the specified macro.	Server contains eight macros. You are trying to load a ninth macro.	Delete a macro before loading another macro.

Error Messages

Message	Description	Cause	Action
1698—Macro Is Not Loaded	Server was unable to execute the specified macro.	Tried to execute a macro that is not loaded.	Execute the SHOW MACRO command to list currently loaded macros. If the macro is displayed, execute the macro again. Verify that the macro is properly spelled. If not displayed, refer to chapter 4, <i>Managing the Communications Server</i> , for the procedure to manually load the macro.

Notices

Radio Frequency Interference Statement

This equipment has been certified to comply with the limits for a Class A computing device, pursuant to Subject J of Part 15 of FCC rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class A limits may be attached to this terminal server. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Instructions to the User

This equipment generates and uses radio frequency energy and if not installed and used properly, i.e., in strict accordance with the operating instructions, reference manuals, and the service manual, may cause interference to radio or television reception. It 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 residential installation or in a commercial environment.

You can test whether or not this equipment causes interference to radio or television reception by turning the equipment on or off. If it is causing interference, you can try to correct the interference by one or more of the following measures—

- Re-orient the receiving antenna
- Relocate the equipment with respect to the receiver
- Move the equipment away from the receiver
- Plug the equipment into a different outlet so that the equipment and receiver are on different circuits

Notices

- Ensure that adapter connections are tightly secured
- If you use peripherals with the equipment that are not offered by the vendor, it is suggested that you use shielded, grounded cables with in-line filters if necessary.
- Consult your dealer service representative if necessary

The vendor is not responsible for any radio or television interference caused by unauthorized modifications to this equipment. It is your responsibility to correct such interference.

WARNING: To comply with the FCC regulations on electromagnetic interference for Class A computing equipment, all cables, including V.35, V.11 and RS-232 cables, must be properly shielded and grounded. Using substitute cables that are not properly shielded and grounded may result in violation of the FCC regulation.

Index

A

- Adding Links to the Server
 - Choosing a type 2-7
 - SLIP links 2-9
- Address Compression 2-15
- Async Map
 - Creating a 2-18
 - Masking 2-18
- AUI Cable 4-2
- Authentication Protocol 2-14
- Auto-Ethernet Selection
 - Setting up 4-2
 - Thinwire A-2
- Autoboot 4-2, 4-3
- Autoconnect 2-47
- Autoinit A-1

B

- Back of server
 - illustration of 1-9
- Backup Configuration Files
 - See *Storing*
- Backwards Command 3-18
- Battery
 - discharged NVRAM battery,
and error message 3-7
- Boot Sources
 - Card 4-31
 - MOP 4-30
 - ROM 4-30
 - TFTP 4-31
- Booting
 - See *Smart RAMCARD/F*
 - See *Smart RAMCARD/S*
- BOOTP Protocol 1-2
 - For host-to-server uploads 4-34
- Break Key 2-90, 3-19
- Broadcast
 - and exiting screen to return to
initial menu 3-15

- Broadcast (Cont'd)
 - and not receiving when
executing diagnostics 3-14
- Broadcasts
 - and receiving when executing
commands 3-14
 - and receiving when LCD Screen
is dark 3-14
 - See *Commands*

C

- CARS (Centralized Account
Reporting System)
 - Altering parameters 4-3
 - Overview 1-4
 - Viewing parameters 4-3
- Circuit Timer
 - Setting to Accommodate the
Transfer Rate 4-7
- Commands
 - about the hierarchical levels 3-8
 - advancing to the next
heirarchical level 1-6
 - and accessing subsequent
hierarchical levels 3-9
 - And executing by using the
[ENTER] key 3-13
 - and exiting the Events Log
display 3-13
 - and level four, variable 3-8
 - and level one, root operator 3-8
 - and level three, characteristic
3-8
 - and level two continued, object
modifier 3-8
 - and level two, object 3-8
 - and level two, operator 3-8
 - and viewing Broadcasted
messages 3-14
- Backwards 3-18
- Change Address 2-86, 4-37
- Change CARS 4-3

Commands (Cont'd)

- Change Domain 2-88
- Change IP Pool 4-8
- Change Port 2-47
- Change Route 2-87
- Change Security 2-94
- Change Server Autoboot 4-37
- Change Server Bootserver 4-38
- Change Server Export 4-37
- Change Service 2-47
- Change/Define Server Autoboot 4-2
- Change/Define/Set Server Ethernet 4-2
- Clear Events 2-86
- Clear IP Pool 4-8
- Connect Rlogin 2-88
- See DEL
- See DIR
- Disconnect 3-19
- See *DOS commands*
- downloading from the front panel 1-7
- downloading root operator using control key 3-12
- Downloading the object level selection 3-13
- Error message from improper execution of 3-10
- Executing and using the [DELETE] key 1-7
- Execution of and using the [ENTER] key 1-7
- Finger 3-17
- Forward 3-18
- Forwards 3-18
- Help 3-16
- hierarchical structure of, explained 3-8
- Level one root operator screen displays 3-9, 3-10
- See *LOAD SLOT*
- Local Switch 2-48
- Logout 3-16, 3-19

Commands

- Moving backward within current command level 1-6
- moving through the root operator menu screens 3-11
- moving to the previous hierarchical level 3-12
- See *REN*
- Resume Session 3-19
- Returning to the previous hierarchical level 1-6
- Selecting a root operator using control keys 3-11
- selecting alpha characters or numerals 1-6
- Selecting the object 3-12
- Server Autoboot 4-2, 4-3
- Set Time 4-42
- shifting between hierarchical command levels 1-7
- Show CARS 4-3
- See *Show Configuration*

Command

- Show IP Pool 4-8
- Show Port Characteristics 2-48, 2-49
- Show Security 2-94
- Show Service 3-16, 3-17
- Show Service Characteristics 2-48, 2-50
- Show Service Characteristics 2-49
- Show Session 3-18
- Stop Interface A-1
- Telnet Enable 2-106
- Timeserver 1-5
- viewing the object level of the Show command 3-12
- Virtual Enable 2-106

Commands option

- choosing from the initial menu screen 3-8

Communications Platform

Chassis

- about 4-17

Communications Platform
 Chassis (Cont'd)
 loading configuration file,
 procedure 4-20
 procedure for saving
 configuration file 4-19, 4-20
 See also *Show Configuration Command*
 slot numbering for ENIC and
 Line Cards 4-17
 Compressed SLIP 2-64, 2-83
 Configuration 2-2
 Via Parser Commands 2-20
 Configuration file types
 See *ENIC*
 See *Intelligent Line Card*
 Configuration Files
 backup 4-16
 backup, overview 4-16
 retrieval, overview 4-16
 Configuration software image
 onboard booting image 3-2
 Configurator
 Before you begin 2-2
 First time 2-5
 Main menu 2-6
 Reconfiguration 2-6
 Screen layout 2-3
 Starting 2-4
 Configurator Parameters
 Address Compress 2-15
 Async map 2-18
 Async Map Mask 2-18
 Authentication Protocol 2-11
 FCS 2-15
 Link Authentication 2-16
 Link Protocol 2-7
 Maximum Receive Unit
 ACTUAL 2-17
 Maximum Receive Unit MAX
 2-17
 Maximum Receive Unit MIN
 2-17
 Port List 2-7
 Proto Compress 2-15

Configurator Parameters
 Quality Monitor 2-14
 Serial connection speed 2-9,
 2-11, 2-14
 State 2-16
 Connect SLIP Command 1-3
 Connecting to RAF Via
 TELNET 3-16
 Control keys, front panel 1-6

D

Data Transparency 4-4
 Diagnostic
 testing channel memory
 interface 3-4
 Diagnostics
 at startup 3-2
 Choosing from initial menu
 screen 3-2
 enable autoboot 3-3, 3-5, 3-6
 re-boot 3-3, 3-5, 3-6
 show channels 3-3, 3-5, 3-6
 show NVRAM 3-3, 3-5, 3-6
 showing channels (Line Cards
 in slots 1 - 4) 3-7
 test Line Card 3-3, 3-5, 3-6
 test NVRAM 3-3, 3-5, 3-6
 test RAM 3-3, 3-5, 3-6
 using re-boot to return to main
 menu 3-7
 Disconnect Command 3-19
 Disconnecting
 from Local Mode 3-19
 from Service Mode 3-19
 DOS Commands
 and managing Smart
 RAMCARD files 4-25
 Dynamic SLIP 2-61, 2-80

E

ENIC Configuration File
 overview 4-16
 Error Codes E-1

Ethernet
and monitoring, using E-NIC
LEDs 1-7
Events log
accessing via Show Events
command execution 3-11

F

FCS (Frame Checking
Sequence) 2-15
File Transfer
Configuring for 4-5
Finger 1-2
Flow Control 2-50
Flow Control, Printer 2-49
Frame Checking Sequence
(FCS) 2-15
Front Panel
and disabling the LCD Screen
1-11
and downloading commands 1-7
and only access to server
diagnostic routines 1-9
and using number or alphabetic
characters 1-6
Control keys, functions of 1-6,
1-8
error message display 3-10
Functions of Keys 1-6, 1-8
Setting up server operations
from, overview 3-1
Front Panel LCD screen
and display parameters of 1-9
See *LCD Screen*
Fundamentals 3-16

H

Help Command 3-16
Help Files in All Protocol
Versions 1-4

I

IBM Terminal Types 2-97
Initial menu screen 3-2
and choosing Diagnostics or
Commands options 3-2
Initialization software 3-2
Initializing the server
and self-text period 3-2
and use of on-board software 3-2
Intelligent Line Card
Configuration
overview 4-16
IP Pool
Adding entries to 4-8
Overview 4-8
Removing entries from 4-8
IPX Netware Printing
Configuring for IPX 2-23
Printing Via 2-30
Understanding IPX Protocol
2-22

K

Keyboard Mapping Profiles
2-98, B-1
Error Codes E-2

L

LAT Service
Connecting to 3-16
Creating Local LAT Services
2-47
Custom Configurations 4-42,
4-43, 4-44, 4-45
LAT Specifications 1-16
LCD Screen
and disabling its display 1-11
and display of Events Log 3-13
and display parameters of 1-9
and lighting up when message
is received 1-9

- LCD Screen (Cont'd)
 - and lighting when front panel key is pressed 1-9
 - and non-function if E-NIC is not installed 1-9
 - and only storing twenty-one lines of text 1-9
 - compared to full-view monitor display 1-11
 - navigating to view other regions of display 1-10
 - retaining information after screen darkens 1-9
 - See also *Semaphore*
 - See also *Subscreen*
- Line Card
 - 3270 - loading configuration 4-23
 - See *3270 Cluster Controller*
 - 3270, loading configuration, procedure 4-24
 - and determining which card is in each slot 3-7
- Line Card testing
 - and slot number to be entered in screen 3-4
 - and testing channel memory interface 3-4
- Line Cards
 - Revision numbers of cards residing in server 3-7
 - Smart RAMCARDS/s 4-16
- Link Authentication 2-16
- Link Destination IP Address 2-10, 2-20
- Load Slot Command
 - See also *Load Slot/Save Slot Commands*
- Load Slot/Save Slot Commands 4-17
 - syntax of 4-18
- Local Cache
 - Setting Host Names and IP Addresses in 2-88
- Local Mode
 - Disconnecting from 3-19
 - Resuming a Session from 3-19
 - Returning to from a Session 3-19
 - Logging into Server 3-16
 - Logging out of Server 3-16
 - Logout Command 3-16
- M**
 - Macro - Menu Facility
 - Automatically Activating Macros 4-11
 - Displaying Macros 4-15
 - Examples of Macros 4-10
 - Manually Executing a Macro 4-14
 - Manually Loading Macros 4-13
 - Writing a Macro 4-9
 - Macro Function 1-4
 - Magic Number 2-14
 - Main diagnostics menu screen
 - accessing of 3-2
 - illustration of 3-2
 - Masking
 - 255.255.255.0 A-2
 - Async map 2-18
 - Telnet Security 2-87
 - Maximum Receive Unit
 - ACTUAL 2-17
 - Maximum Receive Unit MAX 2-17
 - Maximum Receive Unit MIN 2-17
 - Modem
 - Configuring Server for
 - Dial-in/Dial Out Modem 2-54
 - Features 1-4
 - Setting Up a Dial-out Modem Pool 2-53
 - Using BREAK with 3-19
 - MOP Protocol
 - For host-to-server uploads 4-32

MOP Protocol (Cont'd)
 For server-to-server uploads
 4-36
Multisessions
 Understanding 2-101

N

Nameserver
 Adding Access to 2-85
Network Protocol Translation (NPT)
 Connecting to translated services 2-105
 LAT-to-Telnet translation 2-103
 Overview 2-101
 TCP port 23 2-106
Non-LAN Host 2-56
 Using the Server as a Front End 2-56
NVRAM
 and execution of diagnostic test 3-3
NVRAM battery
 discharged battery, possible cause of error 3-7
NVRAM test
 and testing permanent (non-volatile) memory 3-6

O

On-line Help
 Accessing 3-16
 Using 3-16

P

Packet Size Limits 2-17
Packet Types 1-16
PAP/CHAP 1-2
 Overview 2-65
 Sample PAP/CHAP Configuration 2-66

Password
 Host Login 2-89
 Remote Console 2-90
 See also Service
Permanent memory
 Testing of (NVRAM) 3-6
Port
 Configuring for File Transfers 4-5
Power switch
 and powering on the server 3-2
 location and use of 1-7
PPP (Point-to-Point Protocol)
 State of link 2-16
PPP (Point-to-Point Protocol) Links
 Creating links with the Configurator 2-11
PPP Link Setup
 Via Parser 2-57
Printer
 IPX 1-2
Printers
 XON/XOFF Flow Control 2-49
Procedures
 loading 3270 Line Card configuration 4-23, 4-24
 loading communications platform configuration 4-20, 4-21
 loading X.25 PAD Card configuration 4-22
 saving 3270 Line Card Configuration 4-23
 saving communications platform configuration 4-19, 4-20
 saving X.25 PAD Card configuration 4-21
 saving X.25 PAD Card configuration file 4-21
Protocol Compression 2-15

Providing LAT Access to TCP/IP
Telnet Hosts
 LAT Service to Translate to
 Telnet Host 2-103, 2-104
Proxy RIP 2-76

R

RAM
 and testing from the main
 diagnostics menu 3-4
Re-boot 3-7
Re-initializing the server
 due to executing NVRAM
 test 3-6
Remote Console Port
 Connecting to Via Telnet 2-90
Resume Command 3-19
Reverse TELNET 1-2
RIP Interface
 Configuring for 2-72
 Proxy RIP 2-76
 Understanding 2-75
 Understanding 1-3
Rlogin 1-4
 Understanding 2-88
ROM Card
 and downloading software
 booting image 3-2
Routes, Creating 2-87
Running Startup Diagnostics
3-2

S

Save Slot Command
 See also *Load Slot/Save Slot
 Commands*
 storing Intelligent Line Card
 configuration 4-17
 storing ENIC configuration 4-17
 storing standalone server
 configuration 4-17
Security
 Telnet, Setting 2-91

Semaphore
 and moving to display other
 regions of screen 1-11
 illustration of, with comparison
 to monitor 1-11
 Indicating display area of
 message on LCD 3-15
Serial Connection Speed 2-9,
2-11
Serial Line Internet Protocol 1-3
 Dynamic 1-3
Serial Line Internet Protocol
(SLIP)
 Compressed 2-83
 Compressed SLIP 2-64, 2-83
 Configuring a Dynamic SLIP
 Interface 2-81
 Configuring the Server for SLIP
 2-77
Server
 slots available for Line Cards
 1-13
Server Ipsecure 2-91
Service
 Creating Identifying Message
 for 2-54
 Name 2-47, 2-49, 2-51, 2-52,
 2-53, 2-54, 2-56
 New, Testing 2-48
Service Mode
 Disconnecting 3-19
Service Password 3-17
Session Control
 Connecting to an Additional
 Session 3-18
 Disconnecting from the Local
 Mode 3-19
 Returning to Local Mode from a
 Session 3-19
Sessions
 Changing Between 3-18
 Disconnecting All 3-16
 Numbers of 3-18
Set Panel Display Command
1-11

- Show Channels
 - and display of Line Cards
 - residing in server 3-7
 - Show Events
 - and executing to access Events
 - log 3-11
 - Show NVRAM 3-3
 - Show NVRAM screen
 - and pressing any key to return
 - to main menu 3-3
 - Show Service Command 3-16, 3-17
 - Show Session Command 3-18
 - Simple Network Management Protocol (SNMP) 1-3
 - SLIP (Serial Line Interface Protocol)
 - Compressed SLIP 2-83
 - Dynamic SLIP 2-80
 - Link setup using parser 2-77
 - Slots
 - in the server, for inserting Line Cards 1-13
 - Smart RAMCARD
 - overview 4-16, 4-17, 4-18
 - Smart RAMCARD/F
 - booting 4-19
 - Smart RAMCARD/S 4-16
 - 3270 Cluster Controller Line Card 4-19
 - commands 4-17
 - communications platform
 - server 4-19
 - See *ENIC*
 - See *Intelligent Line Card*
 - managing files 4-19
 - X.25 PAD Card 4-19
 - Smart RAMCARD/S
 - overview 4-16
 - SmartRAM CARD
 - From card to server upload 4-38
 - Software Uploading 1-4
 - Software booting image
 - and downloading from host 3-2
 - and ROM Card 3-2
 - Software booting image (Cont'd)
 - onboard 3-2
 - Software Uploading
 - Compatibility of versions 4-30
 - Server Exporting 1-4
 - TFTP 1-3
 - Specifications
 - TCP/IP 1-16
 - Stand-alone Server
 - Saving the configuration 4-21
 - Standalone Servers
 - and not containing physical slots 4-18
 - Startup Diagnostics 3-2, 3-3, 3-4, 3-5, 3-6, 3-7
 - test keypad 3-3, 3-5, 3-6
 - Static SLIP 2-78
 - Subscreen and semaphore
 - which marks region of display 1-10
 - See *Semaphore*
 - Switch, power
 - Location and use of 1-7
- T**
- TCP/IP Specifications 1-16
 - Telecommuting 1-4
 - Telnet
 - Configuring for 2-84
 - Connecting to RAF via 3-16
 - Telnet Host
 - Connecting to 3-17
 - LAT Access 2-104, 2-105
 - Outgoing 2-92
 - Telnet Protocol
 - Configuring for 2-84
 - Telnet Security 2-91, 2-93
 - How It Works 2-87, 2-91, 2-97
 - Masks 2-92
 - Security Addresses 2-93
 - Test keypad screen
 - and canceling the test 3-3
 - TFTP Protocol
 - For host-to-server uploads 4-33

TFTP Protocol (Cont'd)
 For server-to-server uploads
 4-37
 Tilde 2-90
 Time and Date
 Altering current 4-42
 Timeserver 4-45
 TN3270 B-1
 Overview 1-3, 2-96
 TN3270 Protocol Support
 Creating KMP's for Different
 Terminals 2-100
 Keyboard Mapping Profiles 2-98
 TN3270 Protocol Configuration
 2-96
 Tracing a Route
 Overview 4-46, 4-47, 4-48, 4-49
 Sample TraceRoute Output 4-49
 Setting Query Parameter 4-47
 Setting UDP Port Parameter
 4-46
 Starting a TraceRoute 4-48
 Transferring Files Between PCs
 and Hosts 4-4, 4-5, 4-6, 4-7
 Twisted-Pair-Link's Status-Indicator LED
 10Base-T Adapter (optional)
 1-11

U

UNIX
 Overview of interface 1-3
 UNIX Command Set
 Activating UNIX Command Set
 4-52
 Upgrading Server Software Via
 Password Keys
 Activating Server Upgrades 4-50
 Removing Server Upgrades 4-51

V

Viewing Server Configuration
 Commands

Viewing Server Configuration
 Commands (Cont'd)
 example of screen display C-11
 VT Terminal Use 2-96
 With KMPs B-1

X

X.25 PAD Card
 loading configuration,
 procedure 4-22
 saving configuration, procedure
 4-21