DEC TCP/IP Services for VMS

digital

Use

Part Number: AA–PC27B–TE

DEC TCP/IP Services for VMS

Use

Order Number: AA-PC27B-TE

June 1992

This manual provides user information for the DEC TCP/IP Services for VMS software.

Revision/Update Information:	This is a revised manual.
Operating System and Version:	VMS V5.4 and V5.5
Software Version:	DEC TCP/IP Services for VMS V2.0

June 1992

This manual was produced by Telecommunications and Networks Publications.

Contents

10 TELNET Commands

11 File Transfer Protocol Commands

Figures

Tables

Preface

This guide tells you how to use the applications provided by the DEC TCP/IP Services for VMS software (formerly known as the VMS/ULTRIX Connection). These applications are as follows:

- TELNET utility
- File Transfer Protocol (FTP)
- Simple Mail Transfer Protocol (SMTP)
- Remote commands
- Remote printing
- Network File System (NFS) Server

Intended Audience

This guide is for experienced VMS users who use the Applications software and for NFS clients who access file systems mounted on the DEC NFS server. The NFS information is also useful to FTP users who use the Applications software to transfer files between VMS-based systems and other systems.

This guide also provides useful information for non-VMS users who access Applications servers or who use the Applications software.

Document Structure

This guide contains the following chapters:

- Chapter 1 introduces you to the DEC TCP/IP Services for VMS product.
- Chapter 2 provides a general overview of the Internet and how it works.
- Chapter 3 explains how to use the TELNET utility.
- Chapter 4 explains how to use the File Transfer Protocol (FTP).
- Chapter 5 explains how to use the Simple Mail Transfer Protocol (SMTP).
- Chapter 6 discusses the remote commands and services.
- Chapter 7 explains how to use remote printing.
- Chapter 8 shows how to use the Network File System (NFS) to share files.
- Chapter 9 outlines the differences between VMS and UNIX file structures.
- Chapter 10 contains reference information for the TELNET commands.
- Chapter 11 contains reference information for the FTP commands.
- Appendix A lists TELNET error messages.
- Appendix B lists FTP error messages.

- The Glossary defines terms used in this manual.

Associated Documents

In addition to this manual, you may want to refer to the following related documents:

- The *DEC TCP/IP Services for VMS Installation* describes how to install the DEC TCP/IP software.
- The *DEC TCP/IP Services for VMS Programming* describes how to write Internet network applications using the QIO interface and the VAX C socket interface.
- The *DEC TCP/IP Services for VMS System Management* describes how to manage the DEC TCP/IP software. It includes procedures for controlling, monitoring, and testing the DEC TCP/IP software running on a VMS server, as well as procedures for managing the NFS server, controlling VMS resources, maintaining server performance, and troubleshooting server problems.
- The DEC TCP/IP Services *UCX Command Reference* lists the UCX commands used to manage the DEC TCP/IP software.

The following Digital remote procedure call (DECrpc) documents are included with the DEC TCP/IP Services for VMS documentation:

- The *Guide to the Location Broker*
- The *DECrpc Programming Guide*

You can also order the following book through Digital: *Internetworking with TCP/IP: Principles, Protocols, and Architecture* by Douglas Comer (order number ER–TCPIP–TM–001). This book provides an introduction and overview of Internet concepts. It also explains the various protocols, Internet addressing, and other Internet concepts.

Acronyms

The following acronyms appear in this manual:

ARP	Address Resolution Protocol
ASN.1	Abstract Syntax Notation One
BIND	Berkeley Internet Name Domain
BSD	Berkeley Software Distribution
CFS	Container File System
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IRP	Interval Request Packet
MBUF	Memory buffer
MFD	Master file directory

MIB 1	Management Information Base
MOLD	Managed Object Location Directory
MOM	Internet Managed Object Module
NFS	Network File System
PDU	Protocol data unit
RIP	Routing Information Protocol
RMS	Record Management Service
RPC	Remote Procedure Call
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SVID	System V Interface Definition
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UFS	UNIX File System
UUCP	UNIX-to-UNIX Copy Program
VFS	VMS File System

Conventions Used in This Document

The following conventions are used throughout this manual:

ULTRIX,UNIX	UNIX in this book refers to UNIX Version 4.3 of the Berkeley Software Distribution (BSD). ULTRIX is fully compatible with UNIX BSD Version 4.3 systems.
`Special type`	This special type in command examples indicates system output and user input.
lowercase italics	Lowercase italics indicate variables for which you specify or the system supplies the actual values.
`/usr/smith/work`	In text, each mention of a UNIX command, option, directory, or file is presented in this special type.
host, node	These words are used interchangeably to mean any system connected to an Internet network.
[ ]	Brackets enclose optional values. Do not type the brackets in the command line.
key	A keyname enclosed in a box indicates that you must press the indicated key on your keyboard.
Ctrl/*x*	The expression Ctrl/*x* indicates a control character keying sequence. Press the key labeled CTRL and the appropriate character key simultaneously.
$ TYPE MYFILE.DAT . . .	In examples, a vertical series of periods, or ellipsis, means either that not all the data displayed by the system in response to a command is shown or that not all the data a user would enter is shown.
input-file,...	In command syntax or examples, a horizontal ellipsis indicates that additional parameters, values, or other information can be entered, that preceding items can be repeated one or more times, or that optional arguments in a statement have been omitted.
%	The percent sign (%) is the default user prompt for UNIX systems.

You must press Return in order to execute commands. This step is assumed in command examples and therefore is not always shown in command displays.

1

DEC TCP/IP Services for VMS Overview

The DEC TCP/IP Services for VMS product is a set of software packages that provides a TCP/IP network and Network File System (NFS) environment for users on a VMS system.

With the DEC TCP/IP Services for VMS software, VMS systems can communicate with other systems in a heterogeneous computer environment, allowing VMS system users to:

- Store, retrieve, and share files in a distributed, multivendor environment.
- Establish a virtual terminal connection between the VMS system and other compatible systems in the network.
- Communicate with systems that use the Fiber Distributed Data Interface (FDDI), a ten-fold increase in data transfer speed.
- Increase the efficiency of network management operations.
- Exchange electronic mail with other compatible systems in the network.
- Develop custom client and server applications using programming interface libraries.
- Use existing DECwindows applications and implement DECwindows on new applications.

1.1 DEC TCP/IP Services for VMS Components

The DEC TCP/IP Services for VMS product consists of the following functional components:

- Run-Time — The collection of Internet software that provides the TCP/IP network environment.
- Applications — TCP/IP utilities for file transfer and remote operations, including remote printing.
- DEC NFS — Network File System server software that allows users on client nodes to access VMS and UNIX files on the server node.

DEC TCP/IP software products are based on the client-server relationship and provide both client and server capabilities for most of their software components. The client-server relationship can be established with a single host or multiple hosts in a VAXcluster system.

1.1.1 Run-Time Software

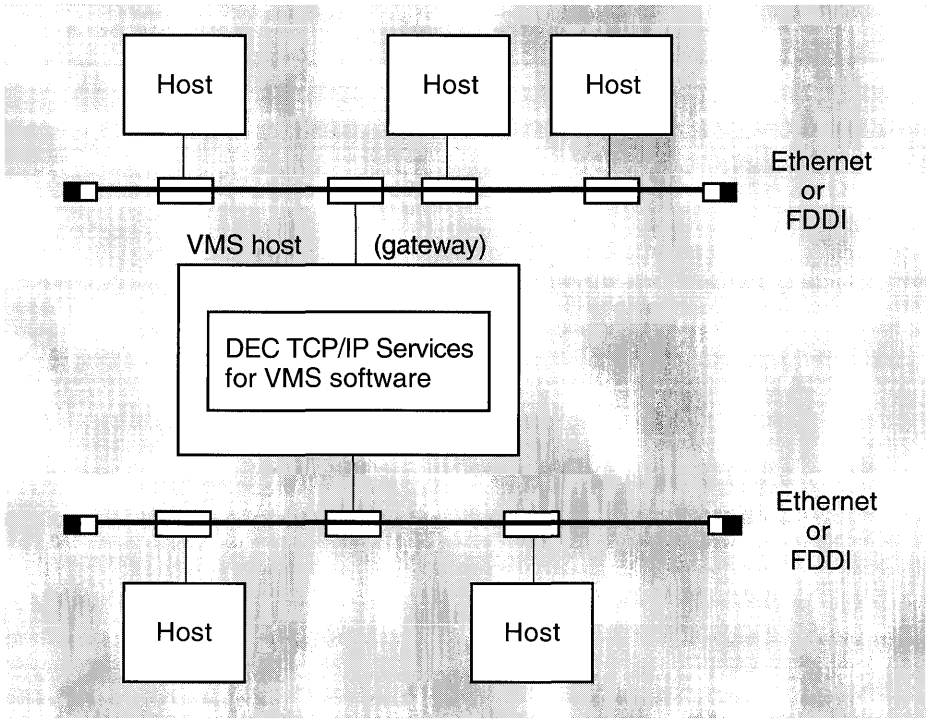
The Run-Time software provides backbone Internet support. Clients communicate with the VMS system by using the Internet networking protocols over an Ethernet or Fiber Distributed Data Interface (FDDI) medium. Communications between the clients and servers are not limited to the local area network; if the local network is connected by a host that serves as a gateway, communications can take place between the local network and other networks.

The Run-Time software includes

- Internet Protocol (IP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)
- Address Resolution Protocol (ARP)
- Dynamic Routing — DEC TCP/IP's implementation of the Routing Information Protocol (RIP)
- DEC TCP/IP Auxiliary server — provides similar functionality as `inetd`
- Berkeley Internet Name Domain (BIND) resolver
- Simple Network Management Protocol (SNMP)
- VAX C socket and QIO programming interfaces

Figure 1–1 shows an example of two networks connected by a gateway.

Figure 1–1 Network Communications



1.1.2 Applications Software

The Applications works together with the underlying network services of the Run-Time to provide the following functions:

- File Transfer Protocol (FTP) — Transfers files between systems.
- Simple Mail Transfer Protocol (SMTP) — Allows you to exchange mail with remote UNIX and VMS systems.
- TELNET utility — Creates a virtual terminal connection to a compatible remote system.
- Remote login — Allows users on a VMS system to log in from their terminal to other systems on the network running a remote login server.
- Remote shell — Allows users to execute a single command, shell script, or command procedure on a remote system without actually logging into that system.
- Remote executive — Allows for command execution on a remote host without actually logging into the remote host. Similar in function to the UNIX `rexec` subroutine call.
- Remote printing — Allows users at VMS or UNIX client nodes to print files on printers attached to VMS or UNIX server nodes.

1.1.3 DEC NFS Software

The DEC NFS software allows users on Network File System (NFS) client systems to access VMS and UNIX-based file systems stored on disks managed by the NFS VMS server.

1.2 What Is the UCX Command Interface?

DEC TCP/IP Services software provides a command interface that allows you to issue commands for any DEC TCP/IP Services component. The command interface is referred to as UCX and is implemented by the VMS image SYS$SYSTEM:UCX$UCP.EXE.

To use the UCX command interface, type the command UCX at the DCL prompt. When the UCX prompt appears, you can issue commands for any DEC TCP/IP Services component installed on your system. For example:

```
$ UCX
UCX> ucx_command
```

where *ucx_command* stands for the command you issue to the DEC TCP/IP Services component.

For more information on DEC TCP/IP commands, see the *DEC TCP/IP Services for VMS UCX Command Reference* manual.

2
Internet Concepts

This chapter introduces the following Internet concepts:

- Client-server model
- Ports
- Internet protocols
- Internet addresses

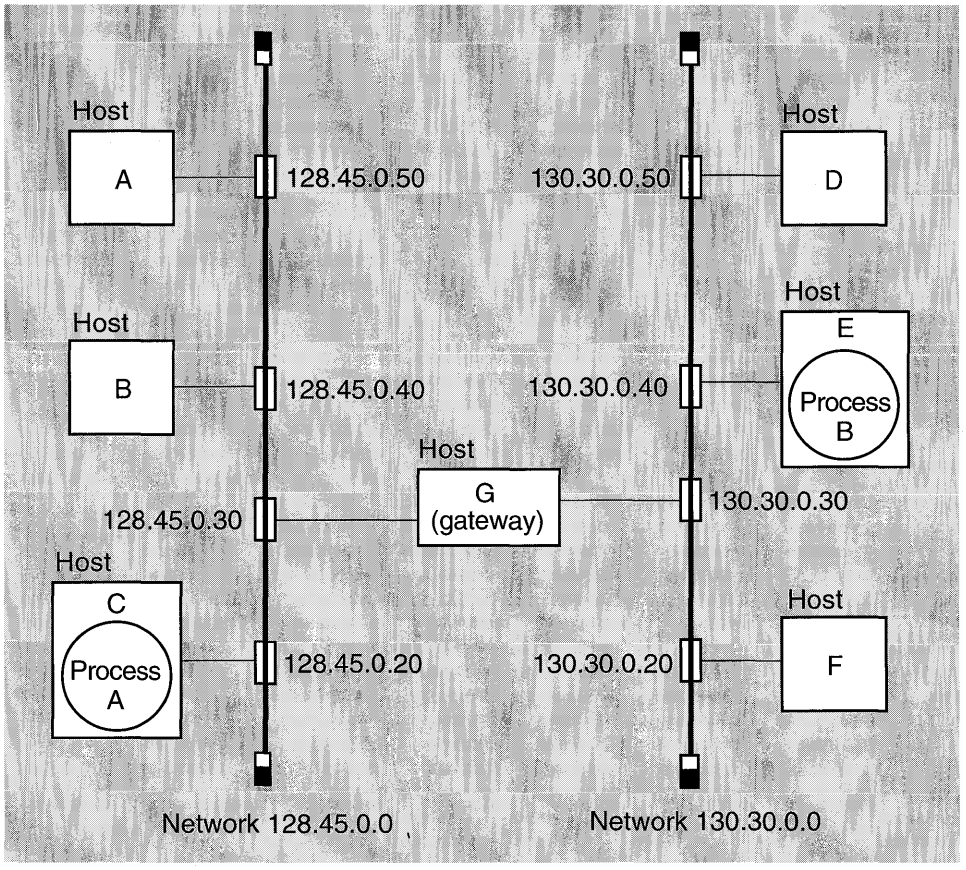
These concepts are necessary for understanding how the DEC TCP/IP Services software works.

2.1 Overview

A **local area network** consists of two or more computer systems connected by an Ethernet or FDDI communication medium. Each host computer connects to the transmission medium by a hardware interface. Each hardware interface is connected to only one local area network.

An Internet network consists of two or more local area networks that are connected by a gateway. A **gateway** (router) is a computer system that transfers data from one computer system to another computer system located on a different local area network. Figure 2–1 shows a possible Internet network configuration.

Figure 2–1 Internet Network Configuration



LKG–5989–92R

Each host in an Internet network is identified by a unique host name and an Internet network address. An Internet network address contains two fields: a network number field and a host number field. Each local area network has a unique network number, and each host on that network has a unique host number.

A DEC TCP/IP VMS server can be a single host; it can also be a whole VAXcluster system or a smaller set of hosts within a VAXcluster system. A VAXcluster system can be represented by a special alias, called the **VAXcluster alias**. A VAXcluster alias appears to identify an actual host to other hosts in the network. Thus, a remote host can address the cluster of hosts as a single host, in addition to being able to address any individual cluster member.

A network can be logically divided into subnetworks. Subnetworks are useful for organizing hosts within a network into logical groups. These logical groups expand the network namespace of an Internet address and allow several logical groups (or logical networks) to exist in the same physical network or networks.

DEC TCP/IP Services products support dynamic routing. If dynamic routing is enabled on your system, gateways on the host network broadcast their routing-database information to other hosts and networks. If dynamic routing is not enabled, you must specify the routes to hosts that are not on your local area network.

2.2 Client-Server Model

Host-to-host communication takes place between two processes. A process is a program that has been scheduled by the system software to execute on a host. A process that offers a service over the network to another process is called a **server**. Servers accept requests from client processes. A **client** sends a request and waits for the result from the server.

A VMS process uses an **Internet pseudodevice** to communicate with the Internet protocols. The Internet pseudodevice contains standard VMS device information. An Internet pseudodevice driver maintains communication-specific information in a structure known as a **socket**.

2.3 Ports

A **port** serves as a destination point for messages transmitted across a network. A message arrives at a particular port, where a process extracts it. Processes waiting at a port are blocked until messages arrive.

To communicate with a process on a remote host, a local process uses both the Internet address and the remote port number of the destination host. The Internet address identifies a particular network and host; the port number identifies the process on that host. Each time an application sends a message, it supplies a port number on the source machine to which replies should be addressed. This makes it possible for the recipient to reply to that message.

For more information on ports, see the *DEC TCP/IP Services for VMS System Management* manual.

2.4 TCP/IP Protocols

The DEC TCP/IP Services software supports the following protocols:

- **Internet Protocol (IP)**

  IP performs two major functions: internetwork addressing and the fragmentation of messages. IP connects various networks and gateways into a system that can deliver data packets from source to destination. Because IP handles addressing information, applications do not need to work directly with network-specific information in order to communicate.

- **Internet Control Message Protocol (ICMP)**

  ICMP is a special-purpose protocol that gateways use to communicate with the network software in hosts. ICMP is a required part of the Internet Protocol.

- **Address Resolution Protocol (ARP)**

  ARP provides dynamic mapping between Internet addresses and Ethernet or FDDI physical addresses.

- **User Datagram Protocol (UDP)**

  UDP provides a datagram mode of communications within the computer network. UDP is used by applications that do not need a reliable stream service. Because UDP does not provide reliable service, some applications add error and sequence control to provide virtual circuits. These virtual circuits improve the reliability of the transmission. The Network File System (NFS) server is an example of an application that uses UDP/IP.

- **Transmission Control Protocol (TCP)**

  TCP is a reliable connection-oriented, end-to-end protocol. It allows two processes to communicate when the processes reside in different host computers that are attached to distinct but interconnected networks. The File Transfer Protocol (FTP) is an example of an application that uses TCP/IP.

- **Simple Network Management Protocol (SNMP)**

  SNMP is based on the network management model of network stations and elements. Network management stations reside in a network operations center (NOC), where they execute management applications that monitor and control the network elements (such as hosts, gateways, and terminal servers). The network elements execute management agents that perform functions requested by the network management stations. DEC TCP/IP implements only the agent portion of the SNMP protocol.

- **File Transfer Protocol (FTP)**

  FTP allows you to log in to a remote host, list remote directories, copy files to or from the remote host, and execute a few simple commands remotely. For more information on FTP, see Chapter 4.

- **TELNET Protocol**

  The TELNET protocol allows you to access any system on your network running the TELNET server software. When you access TELNET, it establishes a virtual terminal connection between your terminal and the specified host. Once a connection is established with a remote host, it appears as if your terminal is connected directly to that host. For more information, see Chapter 3.

- **Simple Mail Transfer Protocol (SMTP)**

  SMTP is a standard protocol that enables users to send and receive mail over the Internet. SMTP specifies the way that the underlying mail system passes messages from one machine to another. The SMTP protocol does not specify the way it accepts mail from the user or the way mail is presented to the user.

  For information on how to use SMTP, see Chapter 5.

- **Berkeley Internet Name Domain (BIND)**

  The BIND service is a host name and address lookup service for the Internet network. The BIND service works according to the client-server model. The client software is referred to as the **resolver**. The resolver allows client systems to obtain host names and addresses from servers rather than from locally hosted databases. Therefore, you can use the BIND service to supplement the host address mapping provided by the local UCX$HOST file.

  Remember that DEC TCP/IP Services only implements the resolver, and that you still need a separate server to use the BIND service. For more information on the BIND resolver, refer to the *DEC TCP/IP Services for VMS Installation* and *DEC TCP/IP Services for VMS System Management* manuals.

- **Routing Information Protocol (RIP)**

  RIP allows gateways to broadcast their current routing database to other hosts and the networks that are connected directly to them.

DEC TCP/IP implements the RIP protocol through its dynamic routing server (UCX$INET_ROUTING.EXE), which runs as a subprocess to the VMS Internet ACP.

2.5 Internet Addresses

When you use Applications, there are many situations in which you must specify the name or Internet address of a remote host. The Internet address is required for the local host to communicate with a remote host.

Internet addressing is described in the *DEC TCP/IP Services for VMS System Management* guide. You may also refer to *Internetworking with TCP/IP: Principles, Protocols, and Architecture* by Douglas Comer. (See the Associated Documents section in the Preface.)

3
The TELNET Utility

This chapter provides the following information about using TELNET:

- Accessing TELNET
- Using TELNET commands
- Switching between input and command mode
- Terminating a session

Chapter 10 provides a summary of TELNET commands and a reference section on these commands.

3.1 How TELNET Works

The TELNET utility allows you to access any system on your network that is running the TELNET server software. When you use TELNET, you establish a virtual terminal session between your terminal and a remote host. That is, your terminal responds as if you were directly connected to the remote host.

If the remote system to which you are connected supports virtual terminals and your session is disconnected for any reason, you can resume the same session when you reconnect to the same host. The system manager of the remote system to which you plan to connect can tell you if that system supports virtual terminals.

3.2 Accessing TELNET

To use TELNET, you must know what hosts are available on your network. You must also have a user account on each remote host you want to access.

The TELNET command has the following syntax:

TELNET [*hostname*] [*port*]

The *hostname* parameter specifies the remote host to which you want to connect. You can use the name, Internet address, or alias of the remote host with this parameter. The *port* parameter specifies a particular communications port on the remote host. In most cases, you need to specify only the *port* parameter if you are connecting to a TELNET server that uses a nonstandard communications port.

There are two ways to start a TELNET session: command mode and input mode.

- Input mode—Connects immediately to the remote host.
- Command mode—Allows you to set up parameters and options on the local host before connecting to the remote host.

You can switch back and forth between command mode and input mode at any time.

3.2.1 Using TELNET in Input Mode

To start a TELNET session in input mode, specify a remote host with the TELNET command. For example:

```
$
```

TELNET logs you in to the remote host ARIEL and requests your user name and password on the remote system:

```
Trying...
Connected to ARIEL.
Escape character is '^]'.

Username:
Password:
ARIEL>
```

After you log in to the remote host, you can issue commands to the remote host as if you were connected directly to it. The remote host will give you its own prompt (in this case, ARIEL>). When you enter commands at the remote host prompt, you are in input mode.

3.2.2 Using TELNET in Command Mode

To use TELNET commands on your local system, you must first be in command mode. TELNET commands let you define the parameters and options for your TELNET session and show session status. When you are in command mode, the commands you enter are interpreted by your local host and passed to the remote host.

To start a TELNET session in command mode, enter the TELNET command at the system prompt. When you enter command mode, the TELNET prompt appears:

```
$
TELNET>
```

To establish a connection with a remote host from command mode, type either `connect` or `open`, and then log in to the remote host. When you establish a connection with the remote host, you automatically enter input mode. In the following example, TELNET attempts to log you in to remote host ARIEL and requests your user name and password on the remote system:

```
TELNET>
Trying...
Connected to ARIEL.
Escape character is '^]'.

Username:
Password:
ARIEL>
```

The Applications TELNET client supports two command interfaces as follows:

- DCL interface
- Standard TELNET command interface

Chapter 10 documents the DCL commands and provides the equivalent standard TELNET command for each DCL command. For more information on the standard TELNET commands, refer to your UNIX documentation.

3.2.3 Switching Between Input Mode and Command Mode

To switch from input mode to command mode, use an escape sequence. The default escape sequence is Ctrl/]. (If you want to change the escape sequence, use the SET ESCAPE command. See Chapter 10 for more information.)

When you enter the escape sequence, the characters you enter are not echoed to the screen, just as if you had entered your password. To return to input mode, press Return after the TELNET command you entered has been invoked. For example:

```
ARIEL>
TELNET>
Connected to remote
Operating in character-at-a-time mode.
Escape character is '^]'.

ARIEL>
```

If you started your TELNET session in command mode, you enter input mode when you establish a connection with a remote host.

3.3 Using TELNET Initialization Command Files

You use a TELNET initialization command file to customize your user environment. The initialization file must reside in your login directory and be named TELNETINIT.INI.

To create an initialization file, create a file with one TELNET command on each line. For example:

```
DISABLE AUTOFLUSH
ENABLE BINARY
ENABLE DEBUG
SET DEVICE/TERMINAL=VT100
SET ESCAPE "^N"
```

In this example, AUTOFLUSH is disabled, binary transmission is enabled, debug is enabled, the terminal type is set to VT100, and the escape sequence is set to Ctrl/N.

3.4 Terminating a TELNET Session

There are two ways to terminate a TELNET session:

- If you entered your TELNET session in input mode, log out of the remote host.
- If you entered the session in command mode, return to command mode and type EXIT or QUIT.

If you are connected to a VMS system, you should always log out of the process on the remote VMS host before you exit TELNET to ensure that the process is properly terminated. If you exit from TELNET while still connected to a remote VMS host, the process may still remain active.

4

The File Transfer Protocol

The File Transfer Protocol (FTP) allows you to perform the following tasks:

- File transfer. You can transfer files between hosts.
- File manipulation. You can create, delete, and rename files.

Applications software provides both client and server support for FTP.

The command procedure UCX$FTPD_STARTUP.COM (located in SYS$MANAGER) starts the FTP server. This procedure is automatically invoked when the Internet network starts up.

This chapter describes how to do the following:

- Access FTP
- Transfer data and print files
- Change the default directory
- Use command procedures and initialization command files
- Use FTP commands over DECnet
- End a session

4.1 Accessing FTP

The FTP command has the following syntax:

FTP [*hostname*] [*port*]

where *hostname* specifies the remote host to which you want to connect. You can use the remote host's name, Internet address, or alias. The value for *port* selects a specific communications port on the remote host. In most cases, you only need to specify the *port* parameter if you are connecting to an FTP server that uses a nonstandard port.

4.1.1 Starting FTP

There are two ways to start an FTP session:

- Issue the FTP command and the host name at the system prompt. This connects you directly to the host and prompts you for your login.
- Issue the FTP command alone at the system prompt. To connect to the remote host, you use a separate CONNECT command.

The following example shows how to start FTP using the FTP command and the host name:

```
$ FTP "harry"
220 harry.xyz.abc.com FTP server Fri Mar 30 23:03:08 EST 1991 ready.
Connected to harry.
Name (harry:username):
331 Password required for username.
Password:
```

The user name defaults to your user name on the local system. If this is your correct user name on the remote system, press Return. If you have a different user name on the remote system, enter the correct user name for that system. The system then prompts you for your password; enter the correct password.

If you did not specify a host name when you typed the FTP command, use the CONNECT command to connect to the remote server. For example:

```
$
FTP>
220 harry.xyz.abc.com FTP server Fri Mar 30 23:03:08 EST 1991 ready.
Connected to harry.
Name (harry:username):
331 Password required for username.
Password:
```

The user name defaults to your user name on the local system. If this is your correct user name on the remote system, press Return. If you have a different user name on the remote system, enter the correct user name for that system. The system then prompts you for your password; enter the correct password.

If the remote system is not a VMS system, it may expect you to enter your user name in lowercase letters. (VMS systems are not case sensitive.) In this case, type your user name in lower case and enclose it in quotation marks, as in `"username"`. (Do not put your password in quotation marks.)

When you establish a session with a VMS host, the VMS host executes your LOGIN.COM procedure. Therefore, you can use your LOGIN.COM command procedure to establish the environment for your FTP session.

4.1.2 Selecting a Command Interface

The Applications FTP client software has two interfaces:

- DCL command interface, based on the VMS operating system
- Standard FTP command interface

To enter FTP from the DCL prompt, type FTP. For example:

```
$
FTP>
```

Because DCL commands are not case sensitive, you may have to enclose file specifications in quotation marks to preserve case sensitivity.

The UNIX-based command interface follows standard FTP command patterns. To use the UNIX-based command interface to enter FTP, define FTP and enter the command as follows:

```
$
$
FTP>
```

The UNIX-based command interface is case-sensitive, so it is not necessary to use quotation marks to preserve case sensitivity. This manual documents the DCL commands and includes the names of the equivalent UNIX-based commands. For a more complete description of UNIX-based command syntax, refer to your UNIX documentation.

To switch from the UNIX-based command interface to the VMS command interface, type `vms`. Once in the VMS command interface, you cannot switch back to the UNIX-based command interface.

For more information on FTP commands, see Chapter 11.

4.2 Transferring Files

When you use FTP to transfer VMS files to a UNIX system, certain record attributes may be lost because UNIX systems do not recognize VMS file attributes. For example, if you transfer an index file from VMS to UNIX and back to VMS, the resulting file's record attributes are lost.

The GET command retrieves a file from the remote host. The PUT command sends a file to a remote host.

To preserve these attributes, use the /FDL qualifier with the GET and PUT commands. This qualifier creates a secondary file that contains the file's VMS record attributes. When a file and its accompanying FDL file are copied from UNIX to VMS, the VMS system uses the FDL file to re-create the correct attributes with the file. (Remember that first you have to PUT the file along with its attributes before you can GET that file.)

For example, the command `PUT/FDL myfile` creates an FDL description file for `myfile`, transfers the FDL file to the remote host, and then transfers `myfile` to the remote host.

The command `GET/FDL myfile` retrieves the FDL file that you created during the PUT operation. The local host creates a new file using the file attributes specified in the FDL file and then retrieves `myfile`, which is then appended to the new file. If you do not specify the /FDL qualifier, the output file organization depends on the transfer type.

4.2.1 FTP File Specifications

When using FTP to transfer files to and from non-VMS systems, place quotation marks around any non-VMS file specification, as in the examples shown in this section.

If you use GET to transfer a file from a non-VMS system, you must specify the remote file specification in the correct syntax for the remote system. For example to transfer the file `/usr/mydir/myfile` from a non-VMS system to a VMS system, use the following command:

```
FTP>
```

If you are already in directory `usr/mydir`, type the following command:

```
FTP>
```

If you use PUT to transfer a file to a non-VMS system, you must specify the remote file specification in the correct syntax for the remote system. For example, to transfer the file MYFILE.TXT from a VMS system to the directory `/usr/mydir` on a UNIX-based system, use the following command:

```
FTP>
```

If you are already in directory usr/mydir, type the following command:

```
FTP>
```

This creates a file called myfile on the remote system.

4.2.2 Using the /LOWERCASE and /NOVERSION Qualifiers

When you use PUT to transfer files from a VMS system to a non-VMS system, the file names are transferred in uppercase with version numbers. This is undesirable on many systems. If you want the file names to be transferred in lowercase and without version numbers, you can use the /LOWERCASE and /NOVERSION qualifiers with the PUT command. These qualifiers ensure that the file is transferred to the remote system in lowercase and without version numbers. For example, the following command transfers the latest version of MYFILE.TXT to the remote system as myfile.txt:

```
FTP>
```

4.2.3 Sending Files to a Printer Using GET and PUT

You can use GET and PUT to send files directly to a printer while in an FTP session. To use this feature, you must specify the actual name of the printer; you cannot use the queue name or logical name. In addition, you can send individual files only to a printer (rather than batches). For example, the following command sends the file MEMO.TXT from the local host directly to the printer LNAX: on the remote host:

```
FTP> PUT MEMO.TXT LNAX:
```

Note that to use this feature with the PUT command, the feature must be supported on the FTP server on the remote system.

The following GET command gets the file Q4EARNINGS and sends it directly to the printer LPA0:

```
FTP> GET Q4EARNINGS LPA0:
```

4.2.4 Using Wildcards

The DCL FTP command interface allows you to use standard VMS wildcards in the following commands:

- DELETE
- DIRECTORY
- GET
- PUT

The wildcard characters recognized by FTP are as follows:

- The percent sign (%) to represent individual characters
- The asterisk (*) to represent multiple characters

If you are using FTP to transfer files to and from non-VMS systems, you must know the file specification conventions for those systems.

If you use wildcards to transfer files to non-VMS systems, you cannot specify a remote file specification. Because files are transferred in uppercase with version numbers by default, you may want to use the /LOWERCASE and /NOVERSION qualifiers during wildcard transfers.

4.2.5 Specifying the Data Type

The SET TYPE command specifies the data type for data transfer. There are two kinds of data types:

- ASCII (the default), which is transferred in 8-bit NVT_ASCII representation. This type transfers data character by character. The resulting file is a sequential file in the VMS Stream_LF format. Select this type when transferring ASCII files.
- IMAGE type, which is transferred as contiguous 8-bit bytes. The resulting output file has fixed length records of 512 bytes. Select this type when transferring non-ASCII files, such as executable image files.

For example, to specify the IMAGE data type, issue the following command:

```
FTP>
```

4.2.6 Specifying the Mode

The SET MODE command specifies the mode for data transfer. Applications software supports only STREAM mode. STREAM mode transmits the data as a stream of bytes.

4.2.7 Specifying the File Structure

The SET STRU command specifies the structure of a file for data transfer. Once this command has been sent to the FTP server, any subsequent GET or PUT command will transfer the file in the specified structure. Applications software supports FILE and RECORD structure.

- FILE structure (the default), has no internal structure. A file is a continuous sequence of data bytes. A DIR/FULL command for the file will show the record format field as STREAM_LF.
- RECORD structured files are sequential with variable records. A DIR/FULL command for a file that was written by FTP will show the record format field as variable length.

You can use the FTP DIR/FULL command to check the structure of a file before and after using the SET STRU command. Refer to your RMS documentation for details on record formats for VMS.

For example, to copy MYFILE in RECORD structure, issue the following commands:

```
FTP>
FTP>
FTP>
```

In this example, both the local and remote hosts are VMS systems. The DIR /FULL of MYFILE will show the record format field to be variable length.

Not all systems support the RECORD file structure. If a UCX FTP client on a VMS system sends a SET STRU command to an FTP server that does not support RECORD structure, the SET STRU command is ignored.

If the server does not support the RECORD file structure and you want to copy a file in RECORD structure from a UNIX system to a VMS system, you can do the following:

1. FTP to the VMS system.

2. Enter a `quote stru r` command. The `quote` command will send `stru r` to the FTP server on the VMS system to let the FTP server set the record structure.
3. Enter a `put` of the file on the UNIX system to the VMS system.

For example, to transfer `myfile` from `abyss` (an ULTRIX system) to LASSIE (a VMS system) in RECORD structure, enter the following commands:

```
abyss>
ftp>
ftp>
```

4.3 Changing the Default Directory

To change the current directory on either the remote or the local host, use the SET DEFAULT command. For example, to change your working directory on a UNIX-based remote host from `/usr/staff/smith` to the subdirectory `/usr/staff/smith/work`, type the following command:

```
FTP>
```

To set the directory to `/usr/staff/jones`, use the following command:

```
FTP>
```

You can also use the following command to set the default directory to `/usr/staff/jones`:

```
FTP>
```

To change the default directory on the local host, specify the /LOCAL qualifier with the SET DEFAULT command. For example, to change from directory DUA0:[SMITH] to DUA0:[SMITH.WORK], type the following command:

```
FTP>
```

4.4 Using FTP from a Command Procedure

You can access FTP from a command procedure. However, when you do this, the CONNECT command does not prompt you for a user name and password. Therefore, you must include an explicit LOGIN (user name and password) in the command procedure. This is also true if you access FTP in batch mode.

The following example shows a sample command procedure file called STRT_FTP.COM . Enclose the username in quotation marks to preserve case. Do not enclose the password in quotation marks.

```
FTP
CONNECT "harry"
LOGIN=(USER_NAME="myusername",PASSWORD=mypassword)
```

To start FTP and initiate the automatic login, type the following command at the system prompt ($): @STRT_FTP.

4.5 Using FTP Initialization Command Files

FTP allows you to use an FTP initialization command file to customize your user environment. Your initialization file must reside in your login directory and be named FTPINIT.INI.

You create an initialization file by creating a file with one FTP command on each line. For example, you could use the following file as an FTP initialization file:

```
ENABLE LOG
DISABLE PARSE
SET DEFAULT "/usr/temp/docs"
SET TYPE IMAGE
```

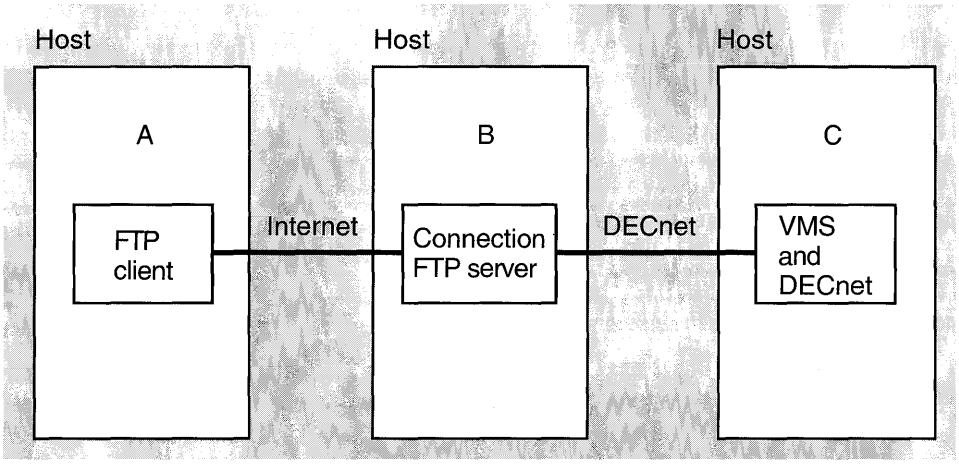
For more information on FTP commands, refer to Chapter 11.

4.6 Using FTP with DECnet

You can use the FTP GET, PUT, and DIRECTORY commands to obtain directory information and to transfer files between FTP clients and DECnet hosts, if the clients and hosts are connected through a host running DEC TCP/IP.

To use FTP to transfer files and obtain directory information in this kind of configuration, you must use the full file specification, including the host, device, directory, and file name. Figure 4–1 shows a sample configuration.

Figure 4–1 Sample FTP/DECnet Configuration



LKG–5950–92R

With this configuration, you could transfer a file from host A to host C with the following FTP PUT command:

```
FTP> PUT MEMO.TXT C::DEVICE$:[DIRECTORY]MEMO.TXT
```

To copy file Q4EARNINGS.TXT from host C to host A, you would issue the following GET command from host A:

```
FTP> GET C::DEVICE$:[DIRECTORY]Q4EARNINGS.TXT MY_Q4EARNINGS.TXT
```

To obtain directory information on C::DEVICE$:[DIRECTORY], you would issue the following command from host A:

```
FTP> DIRECTORY C::DEVICE$:[DIRECTORY]
```

4.7 Ending an FTP Session

To close the control connection and end the FTP session, issue the EXIT command or enter Ctrl/Z.

```
FTP>
$
```

5
The Simple Mail Transfer Protocol

This chapter explains how to use the Applications implementation of the Simple Mail Transfer Protocol (SMTP). SMTP allows you to exchange mail with remote systems within a heterogeneous network environment.

5.1 Accessing SMTP

SMTP uses VMSmail for both its sending and receiving user agents. To send mail using SMTP or to receive mail sent using SMTP, access VMSmail. For example, if you are using a standard Digital terminal, enter the MAIL command at the system prompt.

```
$
```

When you enter VMSmail, the system displays the MAIL prompt.

```
MAIL>
```

5.2 Sending Mail from VMS Using SMTP

To send mail from VMSmail using SMTP, add SMTP% as a prefix to the mailing address, as shown in the following example:

```
MAIL>
TO:
```

The prefix specifies SMTP as the transport protocol. If you use the SMTP% prefix, you can use SMTP as the transport protocol anywhere you can specify a mail address, including FORWARD and CC:.

5.3 Addressing SMTP Mail

Applications implementation of SMTP allows you to use both SMTP and standard VMSmail addresses. Therefore, you can use whichever addressing scheme you prefer.

5.3.1 SMTP Format Rules

SMTP-style addresses allow you to use many different addressing schemes, depending on the destination of your mail and the route it takes. The following list explains SMTP addressing conventions.

- If you are using SMTP format, you must enclose the address in quotation marks.
- In SMTP format, you list the user name first, followed by the host name.
- The user name and host name are separated by an at sign (@).

- Other legal address separators include the following:
  - Percent sign (%), interpreted the same as an at sign (@) if the at sign is not present. Percent signs can also be used to route mail to gateways to non-Internet systems.
  - Single colon (:) indicates the mail is routed through a host.
  - Exclamation point (!) indicates that the mail is routed through a UNIX-to-UNIX Copy Program (UUCP) gateway.
  - Square brackets ([]) indicate an Internet address.
  - Angle brackets (<>) are ignored.
  - SMTP user names are case-sensitive.
  - SMTP host names are not case-sensitive.
  - You can use Internet addresses instead of host names.

5.3.2 Using SMTP-Style Addresses

If you use SMTP addressing schemes to address your mail, you must enclose the address specification in quotation marks. The following examples show how to use SMTP to route mail in a number of different ways.

- Local host

  To send mail to a user on the local host using SMTP, you could address the mail as follows:

  ```
  MAIL>
  To:
  ```

- Remote host

  To send mail to a user on the remote host using SMTP, you could address the mail as follows:

  ```
  MAIL>
  To:
  ```

  Alternatively, you could address the mail as follows using a VMSmail formatted address:

  ```
  MAIL>
  To:
  ```

  Note that in this case you do not need to enclose the host name in quotation marks.

- Routing

  To route mail through a specific host to a user on a remote host, you could address the mail as follows:

  ```
  MAIL>
  To:
  ```

  Another valid SMTP routing addressing scheme is as follows:

  ```
  MAIL>
  To:
  ```

  In VMSmail format the address might look as follows:

  ```
  MAIL>
  To:
  ```

The first host listed is the routing host; the second host is the destination host.

- Internet addresses

  You can only use Internet addresses in SMTP-formatted addresses. You can use Internet addresses anywhere you would normally use a host name. If you use Internet addresses, you must enclose the addresses in square brackets ([]), for example:

```
MAIL>
To:
```

5.3.3 Using Distribution Lists

The Applications implementation of SMTP enables you to send mail using distribution lists. You can use explicit, implicit, and remote distribution lists.

An explicit list uses SMTP-style addresses. It lists the addresses of the users to whom you want to send mail on the To: line. For example, to send mail to an explicit distribution list using SMTP, address the mail as follows:

```
MAIL>
To:
```

An implicit list uses the addresses collected in a file and uses the file name (or logical name) as the address when sending mail. To use implicit distribution lists, enclose the file specification in quotation marks. For example, you could create the file MY_GROUP.DIS that contains the following addresses:

```
smith@abcxyz
jones@xyzabc
ABCDEF::FRANK
```

To send mail to everyone in the distribution list, you could use the name of the file for the mail address as follows:

```
MAIL>
To:
```

The distribution list cannot contain the names of other distribution lists.

SMTP also enables you to use distribution lists that exist on remote hosts. For example, if you are on HOST_X and the distribution list `market_group` exists on `host_y`, you could use the list to send mail by addressing the mail as follows:

```
MAIL>
To:
```

The system manager creates the system-wide distribution list used on the SMTP receiver.

5.3.4 Using the UNIX-to-UNIX Copy Program

The UNIX-to-UNIX Copy Program (UUCP) is an application that allows any system running UUCP to copy files to and from other systems running UUCP. UUCP is usually used to copy files over a dialup connection, which involves connecting the systems by modem. To route mail to a host through UUCP, be certain that the general gateway is defined in the SMTP configuration.

To use SMTP to route mail to a system running UUCP, address the mail as follows:

```
MAIL>
To:
```

5.4 Using SMTP with Other Mail Features

SMTP allows you to use other VMSmail features, such as receiving mail, copying mail, and forwarding mail. The following sections explain how to use SMTP with these features.

- Receiving mail from SMTP

  Receiving mail from SMTP takes place in the same manner as receiving VMSmail. You can use all the features of VMSmail (such as REPLY and FORWARD) to respond to the mail.

- Forwarding mail using SMTP

  To forward mail to other users using SMTP, specify SMTP in the forwarding address. For example, address the mail as follows:

  ```
  MAIL>
  TO:
  ```

- Copying mail to other users

  You can also specify SMTP when you copy mail you send to another user. For example, to send a copy of your mail message to `user_1` on `host1` to USER_2, address the mail the following way:

  ```
  MAIL>
  TO:
  CC:
  ```

- Setting FORWARD using SMTP.

  If you move to a node that uses SMTP as the standard mail protocol, you can specify SMTP as the protocol to forward your mail to your new node. You need to set the remote address within three sets of double quotation marks (as shown in the example) because VMS strips two sets of quotation marks when parsing addresses. For example:

  ```
  MAIL>
  _Address:

  MAIL>
  Your mail is being forwarded toSMTP%"user_name@host"
  ```

5.5 Obtaining User Information

UCX commands allow you to obtain current SMTP information:

- REMOVE MAIL — Deletes the specified mail message from the SMTP queue.
- SEND MAIL — Requeues a mail message for delivery.

For reference information on how to use these commands, see the *DEC TCP/IP Services for VMS UCX Command Reference* manual.

6

The Remote Commands

This chapter explains how to use the Applications implementation of the Berkeley Software Distribution (BSD) remote commands and services.

It describes:

- The remote commands clients and services.
- Requirements for using the remote commands.
- How to use the remote login (`rlogin`) command.
- How to use the remote shell (`rsh`) command.
- How to use the remote execution services (`rexec`).

6.1 Overview of the Remote Commands and Services

The remote commands (called **R commands**) allow sharing of computer resources in a TCP/IP network of trusted hosts. A VMS user can use the services of a remote host by logging in to a remote host or by executing a single command on a remote host. A remote user on a UNIX host can execute a single command or program on the VMS host without logging in to the VMS host.

Table 6–1 summarizes the remote client commands.

Table 6–1 Remote Client Commands

Command	Function
`rlogin`	Allows you to log in to a remote host that provides remote login services.
`rsh`	Allows you to execute a single command, shell script,or command procedure on a remote host without actually logging in to the remote host.
`rsh*`	Invokes the remote execution server (rexecd) on the remote host.

*Using the `rsh` command with the /PASSWORD and /USER_NAME qualifiers invokes the remote execution server. The function is similar to using the UNIX `rexec` subroutine call.

The remote client commands require that the remote host provide the appropriate Applications component services for login and command execution.

Table 6–2 summarizes the remote services.

Table 6–2 Remote Services

Server	Function
rlogind	The remote login server provides a remote login facility with authentication based on privileged port numbers.
rshd	The remote shell server provides command execution services for remote users to execute a single local system command, shell script, or command procedure. Authentication is based on privileged port numbers and user account information.
rexecd	The remote execution server provides remote command execution services for users to execute commands on a remote host. Authentication is based on the server user name and password.

6.2 Remote Command Requirements

The remote client commands require that you set up a trusted host environment allowing access to authorized hosts and users only. The following conditions must be met before you can log in or execute a command on a remote host without entering a remote user name and password:

- You must have a valid user account on the remote host.
- When you use the remote commands from the local VMS host, the name of the local host must be listed in the `/etc/.hosts.equiv` file on the remote UNIX host or the name of the local host and your user name must be listed in the `.rhosts` file in your home directory on the remote UNIX host.
- When you use the remote shell command to execute a command on a VMS system your remote user name and host must be listed in the proxy database. To add an entry to the proxy database, enter the following command at the UCX prompt:

  ```
  UCX>
  ```

  For example, the following command creates a proxy entry for user `minnie` and host `ariel`:

  ```
  UCX>
  ```

6.3 Remote Login

The Applications implementation of remote login provides both client and server functions. If the requirements for the remote commands are met, the functions include:

- Users on a VMS host can log in from their terminals to their account on a remote UNIX system in the network without entering a username and password.

  Users may also log in to a specific account on a remote UNIX system by including a user name with the remote login command.
- Users on a remote host can log in interactively to the VMS host. The VMS host prompts for a user name and password.

The remote host must be running the remote login server.

When you use remote login, it passes the user terminal type to the remote host automatically. Once the connection is established, you have the same privileges and capabilities as though you were connected directly to the remote host.

The Applications component software provides the /LOWERCASE qualifier to convert a VMS user name from uppercase to lowercase before sending it to the remote host. This ensures that an uppercase VMS user name matches its lowercase equivalent on the remote UNIX host. For your convenience, you can add the following line to your LOGIN.COM file:

RLOGIN ::= =RLOGIN/LOWERCASE

6.3.1 Using the Remote Login Client

To access the remote login client, enter the remote login command at the system prompt. You may enter the command with any of the qualifiers described in Table 6–3. The syntax is:

RLOGIN *host_name*

The *host_name* parameter specifies the remote host you want to log in to. You can use the host name, Internet address, or alias to specify the host name.

When a connection is established, you are logged into the remote host and your terminal responds as if it were directly connected to the remote host.

The next section provides examples that describe how to run the remote login command.

Table 6–3 briefly explains the qualifiers you can use with the remote login command.

Table 6–3 Remote Login Command Qualifiers

Qualifier	Function
/ESCAPE_CHARACTER=*escape_char*	Specifies the escape character. Default is the tilde (~).
/LOWERCASE	Converts a VMS username from uppercase to lowercase before sending it to a remote host.
/PARITY=*parity*	Specifies the parity, either 7 (the default) or 8.
/TERMINAL_TYPE=*terminal_type*	Specifies the terminal type if the remote host does not recognize your terminal. Defaults to the terminal type of the terminal you are using.
/TERMINAL_SPEED=*terminal_speed*	Specifies the terminal speed. Defaults to the speed of the terminal you are using.
/USER_NAME=*"user_name"*	Specifies your user name on the remote host. Default is your user name on the local host. Enclose the user name in quotation marks, otherwise it is uppercase.

6.3.2 Remote Login Examples

This section provides examples that describe how to run the remote login command from the client host.

The remote login command requires that you have access to a valid account on the remote host and that you set up either the /etc/hosts.equiv file or the .rhosts file on the remote UNIX host. The examples include how to set up these files.

For each of the examples, user Smith has an account SMITH on a local VMS system LASSIE and an account "smith" on the remote UNIX system ariel.

1. **Remote login to a UNIX host using /LOWERCASE.**

   To log in from the local VMS host to your account on a remote UNIX host, enter the following command at the VMS system prompt:

   For example, for user Smith on host LASSIE to log in to remote UNIX host ariel, the remote login command would be:

   ```
   $
   ariel>
   ```

   For this example, the user account on the local VMS host LASSIE and the trusted host files on the remote UNIX host ariel would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is smith.
     - Add host lassie to the /etc/hosts.equiv file, or
     - Add "lassie smith" to the .rhosts file in smith's home directory.

   The /LOWERCASE qualifier converts SMITH to smith before sending it to the remote host ariel.

   You may also define RLOGIN with the /LOWERCASE qualifier in the LOGIN.COM file as follows: RLOGIN := =RLOGIN/LOWERCASE.

2. **Remote login to a UNIX host without /LOWERCASE.**

   To log in from a local VMS host to a remote UNIX host without the /LOWERCASE qualifier, use the /USER_NAME qualifier and enclose the username in quotation marks.

   For example, for user SMITH on VMS host LASSIE to log in to his account "smith" on host ariel, the remote shell command would be:

   ```
   $
   ariel>
   ```

   For this example, the user account on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is smith.
     - Add "LASSIE SMITH" to the .rhosts file in smith's home directory.

3. **Remote login to a specific user account with /LOWERCASE.**

   To log in to a specific user account on a remote UNIX host, enter the following command at the VMS system prompt:

   ```
   $
   ```

For example, for user SMITH to log in to account "robert" on host ariel, the remote login command would be:

```
$
```

For this example, the user account on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

- **VMS Host LASSIE:** User account is SMITH.
- **UNIX Host ariel:** User account is robert.
  - Add "lassie smith" in the .rhosts file in robert's home directory.

Note

Host names, user names, and passwords on many UNIX (and other non-VMS) systems are case sensitive. Therefore, to preserve case-sensitive host names, user names, and passwords, enclose the case-sensitive item in quotation marks.

4. **Remote login to a specific user account without /LOWERCASE.**
To log in to a specific user account on a remote UNIX host, without the /LOWERCASE qualifier, enter the following command at the VMS system prompt:

```
$
```

For example, for user SMITH to log in to account "robert" on host ariel, the remote login command would be:

```
$
```

For this example, the user account on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

- **VMS Host LASSIE:** User account is SMITH.
- **UNIX Host ariel:** User account is robert.
  - Add "LASSIE SMITH" in the .rhosts file in robert's home directory.

5. **Interactive remote login to the VMS host.**
To log in interactively to the VMS host from a remote UNIX host or a remote VMS host, enter the following command at the system prompt:

For example, to log in to VMS host LASSIE from UNIX host ariel, the command would be:

```
ariel>
LASSIE - Unauthorized access is prohibited
Username:
Password:
```

When a connection is established, the VMS remote system prompts you for a user name and password. After the remote system accepts your password, you are logged into the remote host.

6.3.3 Terminating a Remote Login Session

There are two ways to terminate a remote login session:

- Enter the logout command on the remote host. For example, if you were remotely logged in to an ULTRIX host, you could enter either the `logout` command or the logout key sequence Ctrl/D.
- Enter the specified escape key (or key sequence), followed by a period (.). The default is tilde (~) followed by a period (.).

  To set the escape character to a single character, use the /ESCAPE_CHARACTER qualifier. If you use this qualifier, the escape sequence will be the character you specify followed by a period (.).

6.4 Remote Shell

Remote shell is the Berkeley Software Distribution implementation of the remote shell execution utility. It allows you to execute a command on a remote host while remaining connected to the local host.

Remote shell executes a single command, shell script, or command procedure. Once the command is executed, the control returns to the local host.

Remote execution facilities and authentication are provided by the remote shell service (rshd) and the remote execution service (rexecd).

To execute a command on a remote host, the name of your local host and your user name must appear in the `/etc/hosts.equiv` file or in the `.rhosts` file in your home directory on the remote UNIX host.

To execute a remote command on a VMS host, your user name and host name must be listed in the proxy database. Refer to the *Remote Command Requirements* section in this chapter for more information.

6.4.1 Using the Remote Shell Utility

To access the remote shell client, enter the remote shell command at the system prompt. The syntax is:

RSH *host_name* [/USER_NAME=*user_name*] [/LOWERCASE] *remote_command*

The *host_name* parameter specifies the remote host on which you want to execute a command. You can use the host name, Internet address, or alias to specify the host name.

The *remote_command* parameter specifies the command you want to execute on the remote host.

For example, the following command opens a connection to the remote host `oscar` and executes a `ls` command:

Note

Most commands on UNIX systems are lower case. Therefore, the rsh utility converts all commands to lower case automatically. To preserve case sensitive commands, you must enclose the commands in quotation marks.

Table 6–4 briefly explains the qualifiers you can use with the remote shell command.

Table 6–4 Remote Shell Command Qualifiers

Qualifier	Function
/USER_NAME=*user_name*	Specifies your user name on the remote host. Your user name on the remote host defaults to the same as your user name on the local host. Therefore, you have to use the /USER_NAME qualifier only if your user name on the remote host is different than your user name on the local host.
/LOWERCASE	Converts a VMS user name from uppercase to lowercase before sending it to the remote host. For convenience, you can add the following line to your LOGIN.COM file: RSH :== RSH /LOWERCASE

6.4.2 Remote Shell Examples

This section provides examples that describe how to run the remote shell command.

For each of the examples, user Smith has an account SMITH on the local VMS system LASSIE and an account `"smith"` on the remote UNIX system `ariel`. The following examples describe `rsh` command options for Smith.

1. **Executing a command from the local VMS host with /LOWERCASE.**

   To execute a command from the local VMS host to your account on a remote UNIX host, enter the following command at the VMS system prompt:

   For example, user Smith is logged in to VMS host LASSIE. For Smith to list the files in his current working directory on remote UNIX host `ariel`, the remote shell command would be:

   ```
   LASSIE>
   ```

   For this example, the user accounts on the local VMS host LASSIE and the trusted host files on the remote UNIX host `ariel` would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is `smith`.
     - Add host lassie in the `/etc/hosts.equiv` file, or
     - Add "lassie smith" in the `.rhosts` file in smith's home directory.

For Smith to list the subdirectories of smith's current working directory on remote host ariel the remote shell command would be:

```
LASSIE>
```

The remote command ls -R needs to be enclosed in quotation marks to preserve the case.

2. **Executing a command from the local VMS host without /LOWERCASE.**

   To execute a command from the local VMS host without /LOWERCASE use the /USER_NAME qualifier and enclose the user name in quotation marks.

   For example, user Smith is logged in to VMS host LASSIE. For Smith to list the files in his current working directory on remote UNIX host ariel, the remote shell command would be:

   ```
   LASSIE>
   ```

   For this example, the user accounts on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is smith.
     - Add "LASSIE SMITH" in the .rhosts file in smith's home directory.

3. **Executing a command to a specific remote account with /LOWERCASE**

   To execute a command under a specific remote user name, use the /USER_NAME qualifier with the remote shell command. For example, for user Smith to list the files in robert's directory, the remote shell command would be:

   The user name robert is enclosed in quotation marks to preserve case.

   The user account on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is robert.
     - Add "lassie smith" in the .rhosts file in robert's home directory.

4. **Executing a command to a specific remote account without /LOWERCASE**

   To execute a command under a specific remote user name without /LOWERCASE use the /USER_NAME qualifier with the remote shell command. For example, for user Smith to list the files in robert's directory, the remote shell command would be:

   ```
   LASSIE>
   ```

   The user name robert is enclosed in quotation marks to preserve case.

   The user account on the local VMS host LASSIE and the .rhosts file on the remote UNIX host ariel would be:

   - **VMS Host LASSIE:** User account is SMITH.
   - **UNIX Host ariel:** User account is robert.
     - Add "LASSIE SMITH" in the .rhosts file in robert's home directory.

5. **Executing a command from a UNIX system to a VMS system.**

   To execute a remote shell command from a UNIX system to your account on a remote VMS system, you need to set up a proxy account on the VMS system.

   For example, for user Smith on remote UNIX host `ariel` to display the contents of his home directory on VMS host LASSIE, the user accounts on remote VMS host LASSIE and the local UNIX host `ariel` would be:

   - **VMS Host LASSIE:** User account is SMITH. Set up a proxy entry for Smith on the VMS host as follows:

     ```
     UCX>
     ```

   - **UNIX Host ariel:** User account is `"smith"`.

   The remote shell command would be:

   ```
   ariel>
   ```

6. **Executing a command from a UNIX system to a different account on the remote VMS system.**

   For user Smith to execute a DIR command from UNIX host `ariel` to VMS host LASSIE under the user name robert, the command would be:

   ```
   ariel>
   ```

   For this example, the user accounts on remote VMS host LASSIE and the local UNIX host `ariel` would be:

   - **VMS Host LASSIE:** User account is ROBERT. Set up a proxy entry for Smith on the VMS host as follows:

     ```
     UCX>
     ```

   - **UNIX Host ariel:** User account is `"smith"`.

7. **Executing a command from a local VMS system to a remote VMS system.**

   To execute a command from a local VMS system to the same account on a remote VMS system, add an entry to the proxy database specifying the name of the local host and the user name. Enter the remote shell command at the VMS prompt.

   For example, to execute a `dir` command from local VMS host FRED to remote VMS system, LASSIE, enter the remote shell command at the VMS system prompt on host FRED:

   ```
   FRED>
   ```

   The user accounts on the local VMS host FRED and the remote VMS host LASSIE would be:

   - **Remote VMS host LASSIE:** User account is SMITH. Set up a proxy entry for Smith on the VMS host as follows:

     ```
     UCX>
     ```

   - **Local VMS Host FRED:** User account is SMITH.

8. **Executing a command from the local VMS host under a different user name on a remote VMS host.**

   To execute a command from a local VMS system to a different account on a remote VMS system, add an entry to the proxy database specifying the name of the local host and the user name. Enter the remote shell command at the VMS prompt.

For example, to execute a `dir` command from local VMS host FRED to remote VMS system, LASSIE, under user name ROBERT, enter the remote shell command at the VMS system prompt on host FRED:

```
FRED>
```

The user accounts on the local VMS host FRED and the remote VMS host LASSIE would be:

- **Remote VMS host LASSIE:** User account is ROBERT. Set up a proxy entry for Smith on the VMS host as follows:

  ```
  UCX>
  ```

- **Local VMS host FRED:** User account is SMITH.

6.4.3 Terminating a Remote Shell Command

To stop or interrupt the execution of the command on the remote host, enter Ctrl/C or Ctrl/Y.

6.5 Using the Remote Execution Server

The DEC TCP/IP Applications software provides the command interface `rsh*` to invoke the remote execution server (REXECD).

To invoke the remote execution server, REXECD, on a remote VMS or UNIX host, enter the following command at the VMS system prompt:

RSH *host_name* [/USER_NAME= *"user_name"*]
PASSWORD=*password* [/LOWERCASE] *remote_command*

You must include a password with the remote shell command.

Enclose the user name and password in quotation marks to preserve case.

The following examples illustrate this command:

- To execute a command from the local VMS host FRED to the remote VMS host LASSIE, enter the following remote shell command at the VMS system prompt:

  ```
  FRED>
  ```

- To execute a command from the VMS host LASSIE to the remote UNIX host `ariel` enter the following remote shell command at the VMS system prompt:

  ```
  LASSIE>
  ```

You may also use the `rexec` routine call from a UNIX system to invoke the UCX REXECD server on a VMS host. Refer to your UNIX documentation for more information.

7
Remote Printing

This chapter explains how to use the remote printing feature of the DEC TCP/IP Services for VMS software product.

7.1 Remote Printing Services

The DEC TCP/IP remote printing services allow you to:

- Print local files on printers attached to a remote VMS or UNIX server system.
- Remove jobs from the remote print queue.
- Display status of jobs on the remote print queue.

The remote printing services operate transparently to the user. When you issue a remote printing command from your system, you do not need to be aware that the printer is connected to a different computer system.

DEC TCP/IP remote printing implements the Line Printer Daemon (LPD) Protocol described in the RFC-1179.

7.2 Remote Printing Client and Server Facilities

DEC TCP/IP remote printing services supports client and LPD server facilities for both UNIX and VMS systems. As a user on a VMS or UNIX client system, you can print files on printers attached to a remote VMS or UNIX server system.

VMS client and server systems must run DEC TCP/IP LPD software. UNIX client and server systems must run BSD 4.2 line printer spooler software `lpd`.

The LPD server accepts files for printing, displays information about jobs in the remote queue, and removes jobs from the remote queue.

Table 7–1 briefly describes the client commands.

Table 7–1 Remote Printing Commands on VMS

Command	Function
PRINT	A DCL-style command that allows a user at a VMS client to print one or more files on a printer attached to a server.
LPQ	Displays remote job queue information in long or short form.
LPRM	Removes a job from the remote queue.

Users at a UNIX client use the UNIX-style `lpr` command to print one or more files on a printer attached to a server.)

7.2.1 How Remote Printing Works

From a VMS client system:

The system manager on a VMS host can set a flag in the SYS$SPECIFIC:[UCX_LPD]UCX$PRINTCAP.DAT file that designates a queue as a remote queue and specifies the name of the remote host to which to send the files for printing.

To print a file on the remote printer, the VMS user issues the DCL PRINT command specifying a queue that is mapped to the remote printer through the SYS$SPECIFIC:[UCX_LPD]UCX$PRINTCAP.DAT file. The UCX LPD symbiont that controls the local VMS queues determines the name of the remote host and the name of the remote printer (where the file should be printed) from the UCX$PRINTCAP.DAT file. The job is sent through TCP/IP to the destination host for printing.

From a UNIX client system:

The system manager of a UNIX host can set a flag in the /etc/printcap file that designates a printer as a remote printer and specifies the name of the remote host to which to send the files for printing. See the UNIX printcap documentation for more information.

When a user on a UNIX client system issues a remote printing command, the LPD Server on the VMS host does the following:

- Receives the job request from the client.
- Accepts control and data files from the remote host and stores them in the spool directory associated with the requested queue name.
- Determines the name of the user who submitted the job and verifies that the user is authorized to use the server.
- Determines the name of the local print queue and queues the job to the destination print queue.

See the *DEC TCP/IP Services for VMS System Management* guide for more information.

7.3 Remote Printing from a VMS Client

To print files from a VMS client, use the DCL-style PRINT command:

```
$  PRINT [options] file_spec [,...]
```

Where *file_spec* is the name of one or more files to be queued. All options of the DCL-style PRINT command are supported.

To be notified when your job is completed on the remote printer, add the /PARAMETER=MAIL qualifer to the PRINT command. You will be notified through SMTP mail.

7.4 Remote Printing from a UNIX Client System

To print one or more files from a UNIX client system, issue the UNIX-style `lpr` command:

```
%  lpr [options] [file_spec,...]
```

Where *file_spec* is the name of one or more files to be queued.

All options of the UNIX `lpr` command are supported.

To be notified when your job is finished on the remote printer, add the `-m` qualifier to the `lpr` command. You will be notified through SMTP mail.

7.5 Displaying Print Status

To display information about the status of your job on the remote host, use the DCL LPQ command. Refer to your VMS DCL documentation for the LPQ command syntax.

Table 7–2 briefly explains qualifiers you can use with the LPQ command.

Table 7–2 LPQ Qualifiers

Qualifier	Function
/FULL	Displays status of the remote queue in the long form. The default is NOFULL and the display is the short form.
/USER	Displays status of the user jobs in the remote queue. The default is NOUSER.
/ENTRY	Displays status of the job entries in the remote queue. The default is NOENTRY.

7.6 Removing a Job from the Print Queue

To remove a job from the remote queue, use the DCL LPRM command. Refer to your VMS DCL documentation for the LPRM command syntax.

Table 7–3 briefly explains qualifiers you can use with the LPRM command.

Table 7–3 LPRM Qualifiers

Qualifier	Function
/ALL	Removes all of the jobs from the remote queue for the current user.
/USER	Removes the jobs from the remote queue for the specified user.
/ENTRY	Removes the specified job entries from the remote queue.

8

Using the DEC NFS Component Server

This chapter provides information about file sharing and mounting file systems on the DEC NFS server.

For information about backing up and restoring DEC TCP/IP file systems, see the *DEC TCP/IP Services for VMS System Management* manual.

8.1 Mounting File Systems

The DEC NFS server allows UNIX systems running NFS client software to access data on a VMS host. The NFS server manager makes directories available to NFS clients. The UNIX workstation user (the client) mounts remote directories located on the VMS NFS server onto local directories. The client then accesses files on the VMS server transparently through the local directory.

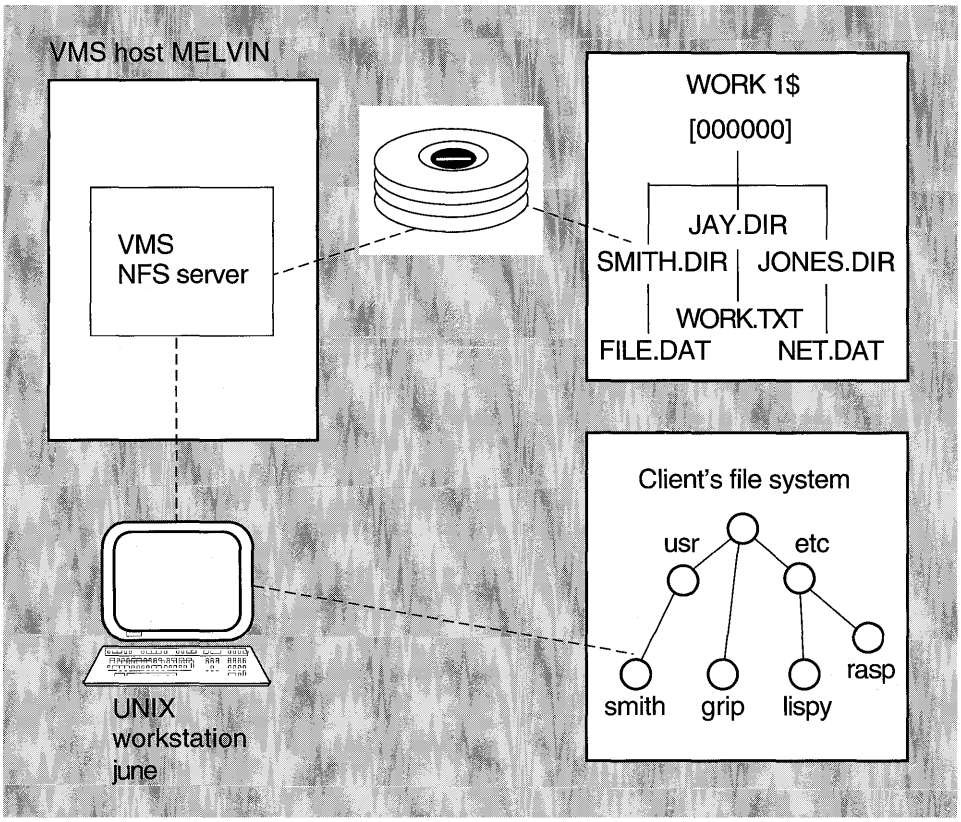
To mount a remote directory from the UNIX client, you use the `mount` command. For example, to mount the VMS file [SMITH.WORK]FILE.TXT located on node MELVIN, you would type the following command from the UNIX client:

```
june>
```

The `automount` command enables you to mount remote file systems without having to specify them using the `mount` command. For more information on the `mount` and `automount` commands, refer to the NFS documentation for your UNIX client.

Figure 8–1 illustrates the way a file system can appear to a client. The NFS server on host MELVIN exports the directory [SMITH] to the NFS workstation `june`. The user mounts the directory WORK1$:[SMITH] onto the workstation's file system. The files in the VMS file system at directory [SMITH] appear to be in the NFS workstation directory `/usr/smith`. If you issue a command to display the local directory `/usr/smith`, you see the file `file.dat`.

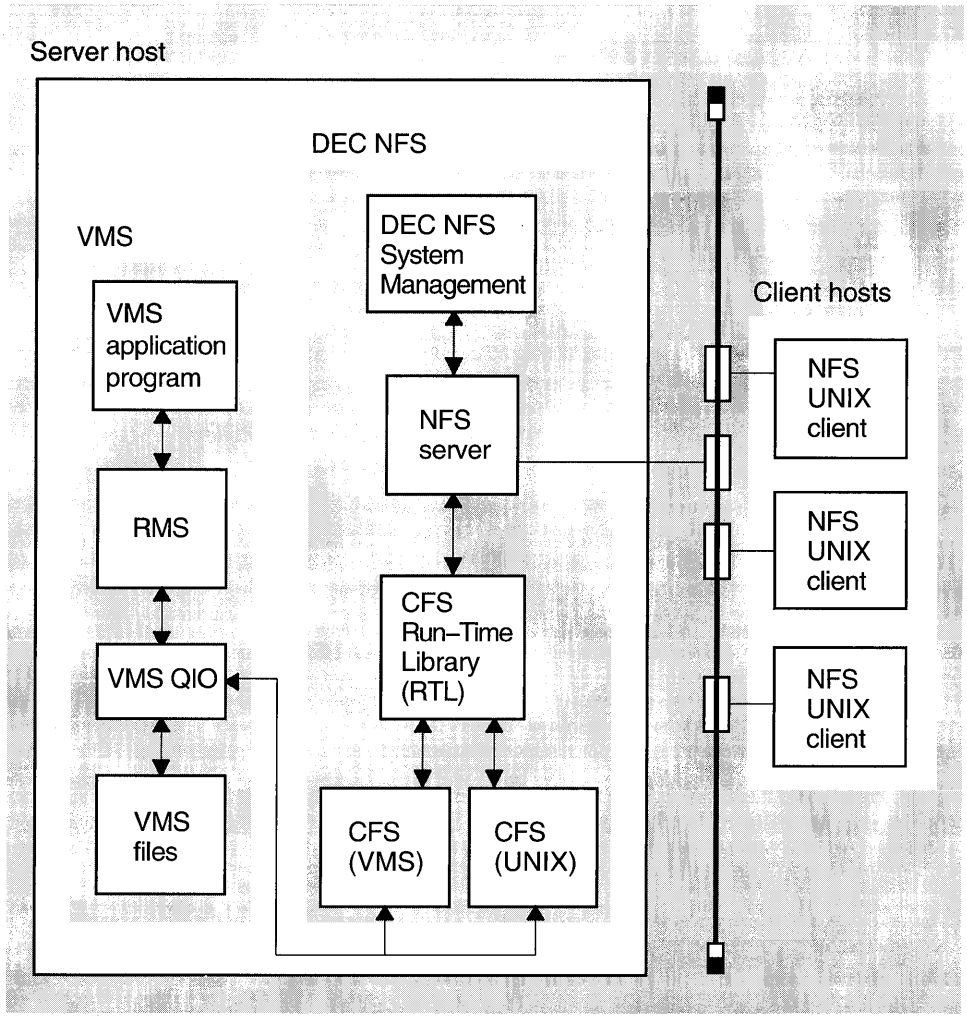
Figure 8–1 A Client's File System



LKG–6001–92R

Figure 8–2 shows how DEC NFS interrelates with the Run-Time and VMS **Record Management Services** (**RMS**). For a description of the components, see Section 5.1.4 in the *DEC TCP/IP Services for VMS System Management* manual.

Figure 8–2 DEC NFS Software Components



LKG–6002–92R

The DEC NFS server supports the `automount` command. The `automount` command allows you to implicitly mount file systems without specifying the `mount` command.

The *DEC TCP/IP Services for VMS System Management* manual contains information on how to prepare file systems for mounting.

8.2 Sharing NFS Mounted Files

The DEC NFS server allows you to simultaneously share files with other NFS clients. Shared access between NFS clients to a DEC NFS-mounted file provides the same behavior as the native NFS environment.

Several users can simultaneously access the same file for read or write operations. Access to the same data within the file is synchronized by the DEC NFS file system, using the VMS lock manager.

The DEC NFS file system does not use RMS to access files and does not share the same locks with RMS. The DEC NFS file system opens files in shared mode. This means that files opened by the DEC NFS can also be accessed by RMS users. However, because they do not use the same locks, there is no access synchronization between a client accessing files through the DEC NFS and local VMS users using RMS for file access.

It is recommended that you do not simultaneously share files mounted through the DEC NFS server with local VMS users because the results may be unpredictable.

8.3 Record Formats

DEC NFS supports the following record formats:

- Fixed length
- Variable length
- Variable with fixed-length control (VFC)
- Stream (including STREAM_LF and STREAM_CR)
- Undefined

Note

Access to non-STREAM_LF files is limited to read-only access.

8.4 Using NFS in a VAXcluster

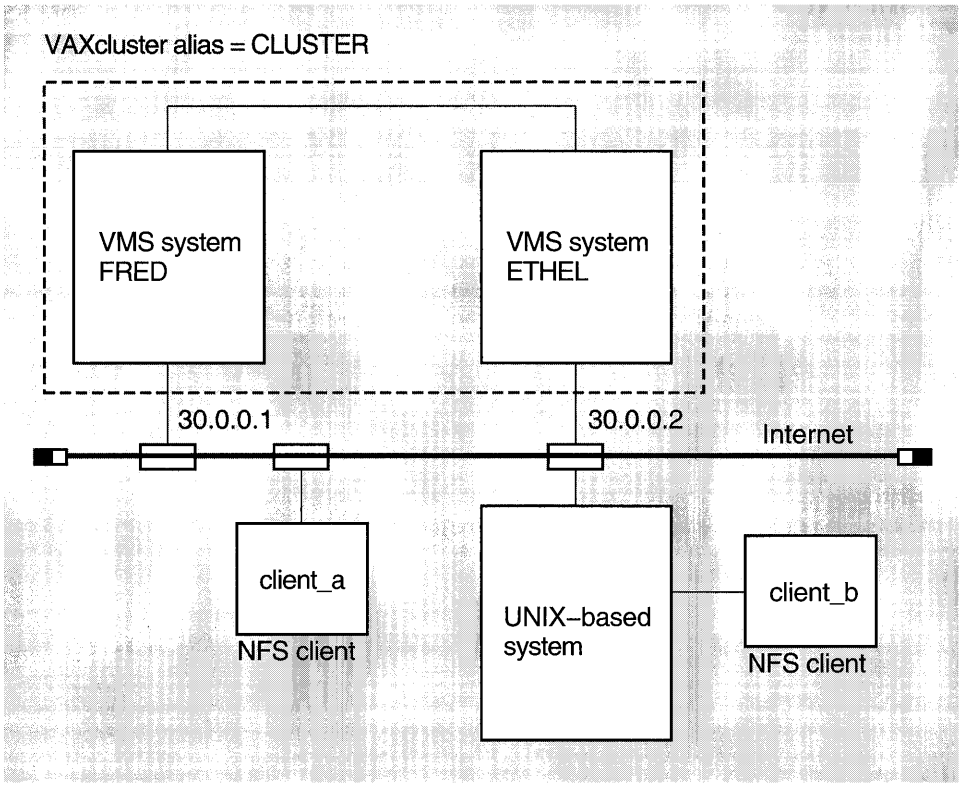
DEC NFS implements the VAXcluster alias feature. A VAXcluster alias uses a single host name to represent all the hosts in a cluster.

To take advantage of this feature, your VMS system manager uses DEC TCP/IP Services for VMS commands to define the cluster alias and Internet interfaces for the cluster. For example, Figure 8–3 represents a sample configuration. This configuration consists of VMS nodes FRED and ETHEL with a cluster alias of CLUSTER.

Once the cluster alias and Internet interface are defined, you can mount file systems from either node and use the cluster alias (`cluster`) in the `mount` command to take advantage of the VAXcluster features.

```
client_a> mount cluster:/mydir/mnt/tmp
```

Figure 8–3 VAXcluster



LKG–6006–92R

For more information, see your VMS system manager and Section 3.6 of the *DEC TCP/IP Services for VMS System Management* manual.

9

Differences Between VMS and UNIX File Systems

This chapter provides information for non-VMS users who will access NFS-mounted file systems on the DEC NFS server. It provides information on the major differences between the VMS and UNIX file systems.

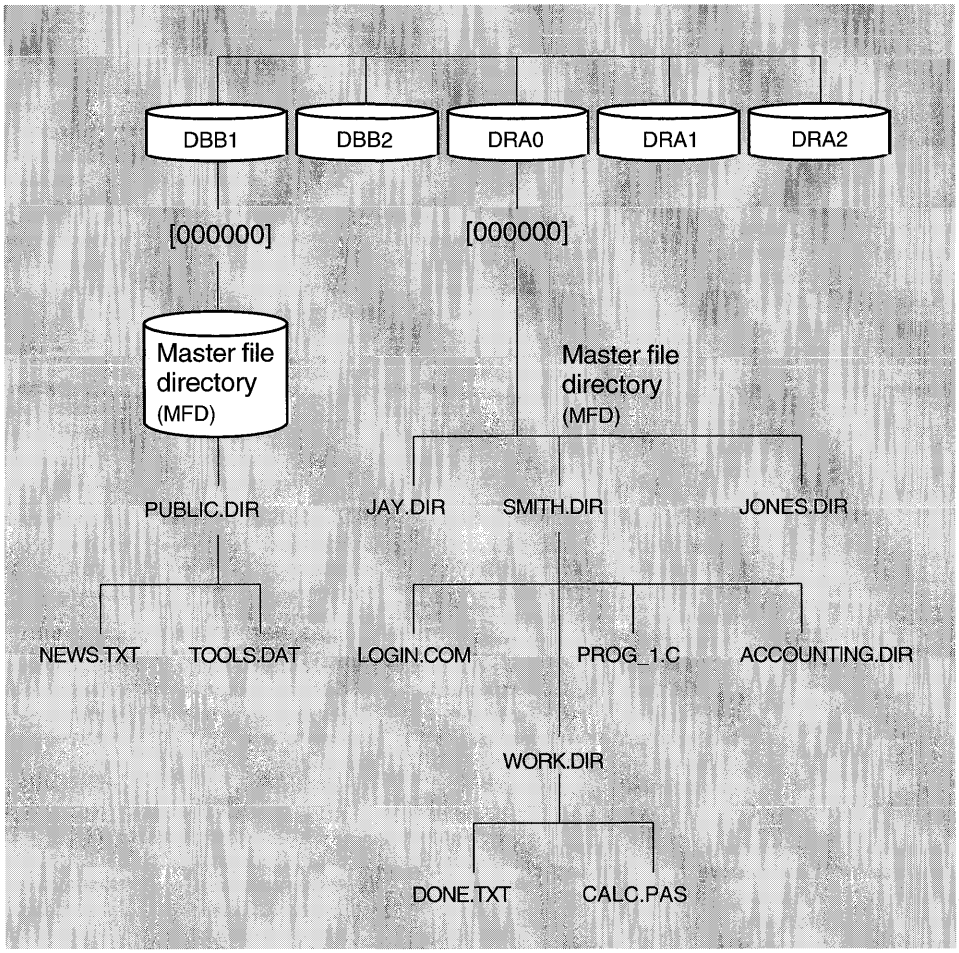
9.1 Directory Hierarchies

The VMS and UNIX system directory structures are similar; the major difference between them is the root directory. The UNIX directory system uses a single root, whereas the VMS directory system uses multiple roots. Each VMS volume has a separate root with all the directories beneath that root contained on that volume. In contrast, the UNIX directory hierarchy has one root, and the directory hierarchy can reside on multiple volumes.

Because the VMS directory structure uses multiple roots, VMS system users must know which device contains their directories and files; UNIX system users do not need to know that information. Therefore, on VMS systems, you sometimes have to include the name of a device in a file specification. On UNIX systems, you never have to include the device name.

Figures 9–1 and 9–2 illustrate directory hierarchies for each operating system. Figure 9–1 shows two VMS directory hierarchies; one is mounted on device DBB1: and the other is on device DRA0:.

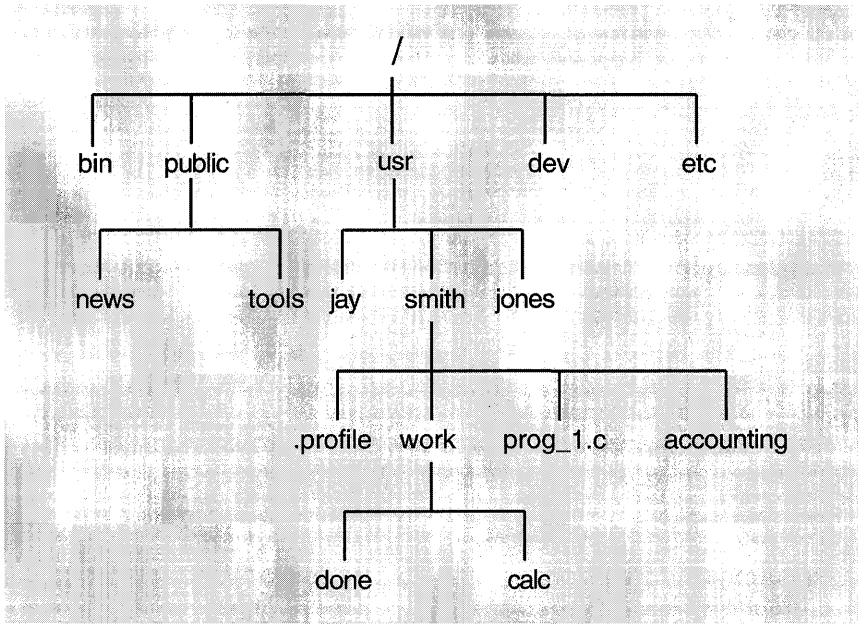
Figure 9–1 VMS Directory Hierarchy



LKG–6397–92R

In contrast, Figure 9–2 shows a UNIX directory hierarchy similar to the VMS hierarchy. Note that the UNIX hierarchy appears as one tree, but it can be located on more than one device.

Figure 9–2 UNIX Directory Hierarchy



LKG–6398–92R

The following sections describe the major differences between the VMS and UNIX file systems.

9.2 File Specifications

There are major differences in the VMS and UNIX file specifications. For example, the VMS file specification for the file DONE.TXT in Figure 9–1 is:

```
DRA0:[SMITH.WORK]DONE.TXT;3
```

In comparison, the UNIX file specification for the equivalent file `done` in Figure 9–2 is:

```
/usr/smith/work/done
```

9.2.1 File Specification Delimiters

A VMS file specification uses the following format:

device:[*directory.subdirectory*] *filename.type*;*version*

Delimiters separate the file specification components. The colon (:) separates the device from the directory. The directory and any subdirectories are contained within brackets ([ ]) or angle brackets (<>). A period (.) separates directories from subdirectories and separates the file name from the file type. A semicolon (;) or period (.) separates the file type from the version number.

A UNIX file pathname uses the following format:

/directory/directory/filename

The slash (/) is the only delimiter that the UNIX file specification format uses. The first slash in a UNIX file specification represents the root directory, and the subsequent slashes separate each component in the file specification (the directories from the other directories and the file name). In theory, there is no limit to the number of directory levels in a UNIX file specification, whereas a VMS file specification is limited to eight levels.

9.2.2 UNIX pathnames

There are two types of UNIX pathnames: absolute and relative. An **absolute pathname** lists the entire directory path that leads to the file, beginning with the root (which is represented by an initial slash). Root is the first directory in the file system. All other files and directories trace their ancestry back to the root. For example, the absolute pathname for the file `calc` in Figure 9–2 is `/usr/smith/work/calc`.

A **relative pathname** begins the directory path with the current process default directory and does not include the current directory name in the pathname. For example, the relative pathname for the file `calc` (in the current directory `/usr/smith`) in Figure 9–2 would be `work/calc`.

A UNIX pathname can have a maximum of 1024 characters. A VMS file specification can have a maximum of 255 characters.

9.2.3 File Names

A complete VMS file name specification includes the file name, the file type, and an optional version number. The file name and file type can each have up to 39 characters and are separated with a period (for example: FILE_NAME.TXT;1). The valid characters in a file name or type include: A–Z, 0–9, underscore (_), hyphen (–), and dollar sign ($). Version numbers are decimal numbers from 1 to 32767; they differentiate versions of the same file.

A UNIX file specification can contain up to 1024 characters. Each element of the pathname can have up to 255 characters, though some versions of the UNIX operating system limit the size of one element to 14 characters. In theory, you can use any ASCII character in a pathname except for the slash (/) and null characters. However, you should avoid using some characters (such as the pipe | character) because these characters can have special meaning to the UNIX shell. For example, a file name of `report_from_january_24` is valid.

As a general rule, it is best to restrict UNIX file names to A–Z, a–z, 0–9, the underscore (_), and the period (.). Do not use a period (.) as the first character in a UNIX file name because the leading periods signify hidden files to the operating system (for example, `.mailrc`). Hidden files do not appear when you issue the `ls` command; to see hidden files, you need to use `ls -a`.

9.2.4 Case Sensitivity

The VMS file system is not case sensitive; the VMS operating system treats all alphabetic characters as uppercase. However, the UNIX operating system treats upper- and lowercase characters as different characters entirely. For example, consider the following file names:

- CHAPTER_ONE
- Chapter_One

- chapter_one

On a UNIX system, these names represent three different files; to the VMS system, they represent the same file. Because the VMS system automatically converts file names to uppercase, put all UNIX file names in quotation marks (" ") when using DEC TCP/IP commands on a VMS system. This prevents lowercase characters from being converted to uppercase.

9.2.5 File Types

File types are an important part of the VMS file name identification. The file type usually describes the kind of data in the file. VMS recognizes a number of special default file types. For example, a MACRO source file typically has a file type of .MAR. Directories all have file types of .DIR;1. (The version number 1 is required on all directories that do not have the /TYPELESS_DIR option.) Files under mount points with the /TYPELESS_DIR option can have the type and version number automatically removed. (See Section 9.6.1.)

In UNIX, directories do not have special file types, since UNIX systems do not use file types. UNIX does use certain naming conventions that resemble VMS file types. For example, file names ending in *.c* are associated with files written in the C programming language; files ending in *.h* are include files (containing header data) for C programs. These naming conventions may be required for a program to compile, but are not imposed by UNIX.

9.2.6 Version Numbers

Every VMS file has a version number. When a file is created, the system assigns it a version number of 1. Subsequently, when a file is edited or additional versions of that file are created, the version number automatically increases by 1. Therefore, many versions of a file with the same file name can exist in the same directory.

If you create a VMS file with an explicit version number and a file already exists with the same file name and version number, the existing file is overwritten.

To locate the most recent version of a file, specify a version number of zero, as in FILE_NAME.TXT;0, or do not specify a version number at all. Specifying version –1 locates the next most recent version, –2 the version before that, and so on.

The UNIX file system does not support automatic creation of multiple versions. In most cases, if you edit a UNIX file, the system saves only the most recently edited copy.

9.3 Links Between Files

The UNIX system allows users to share files by means of links. A **link** is an entry in a directory that refers to a file. A file can have multiple links. The UNIX system maintains a count of the number of links to each file.

There are two kinds of links: hard links and symbolic links.

9.3.1 Hard Links

A **hard link** to a file is indistinguishable from the original link established when the file was created. UNIX users can create additional links to a file with the `ln` command. These additional links allow users to share the same file under different pathnames. Any changes to the file are independent of the link used to refer to the file. A file cannot be deleted (removed) until all links to the file are deleted.

The VMS system also allows you to create multiple links to a file. The following list explains the differences between links in UNIX and links in VMS:

- With VMS you can establish a link to any type of file. UNIX does not allow links to directory files.
- VMS does not maintain a count of links to a file. As a result, a file can be deleted at any time, leaving invalid links to the file. Since UNIX maintains a link count, this cannot happen on a UNIX system.
- VMS files can exist without any links, which means that these files must be accessed by their file IDs. UNIX files cannot exist without links.

You delete a UNIX file by removing all the links to the file. When the last existing link to the file is removed (the link count becomes zero), the information stored in the file is no longer accessible. If there is more than one link to a file, a link can be removed (including the link used to create the file), and the file is still accessible from any remaining links.

All links to a file are of equal value. The UNIX operating system cannot distinguish the order in which links are created.

9.3.2 Symbolic Links

A **symbolic link** is a type of file that contains the name of the file to which it is linked.

A symbolic link can span file systems. Unlike the hard link, the symbolic link does not maintain a link count. Symbolic links can exist after the file has been deleted; however, an error is returned if the symbolic link file is accessed after the file it names is deleted.

The VMS file system does not support the symbolic link function.

9.4 File Structures

The VMS file system supports three file organizations: indexed, relative, and sequential. It also supports the following record formats:

- Fixed length
- Variable length
- Variable with fixed-length control (VFC)
- Stream (including STREAM_LF and STREAM_CR)
- Undefined

Note

Access to non-STREAM_LF files is limited to read-only access. This applies only to files mounted by NFS.

The UNIX file system supports only byte streams. The records in UNIX text files have the same format as the VAX RMS STREAM_LF record format.

Sequential files that are not streams (STREAM_LF or STREAM_CR) can be translated when read. These files cannot be written unless the conversion is disabled or the file is superseded.

9.5 File Protection

The VMS and UNIX operating systems use different file protection schemes. The following sections describe these differences.

9.5.1 VMS File Protection

The VMS operating system uses two mechanisms to control access to files and directories: user identification code (UIC) based protection and access control lists (ACLs).

A UIC is a 32-bit value that consists of a 14-bit group number and a 16-bit member number. Each user of the system has a UIC defined in the SYSUAF file. Access to objects depends on the relationship between the UIC of the process doing the accessing and the UIC of the object (the file or directory).

Users are classified into one or more of the following categories: SYSTEM, OWNER, GROUP, or WORLD. The classification depends on the relationship between the UICs of the accessing process and the object. Each category of user can be allowed or denied any of the following access types: read, write, execute, or delete. (Execute access controls search access to a directory.)

Protection classification takes the following form in VMS:

```
s:rwed,o:rwed,g:rwed,w:rwed
```

ACLs can also control access to files. By using ACLs, you can explicitly deny or grant read, write, execute, delete, and control access to a user or group of users that have the identifier specified by the ACL. See the *VMS System Management Security Access Control List (ACL) Editor* manual for additional information about ACLs.

9.5.2 UNIX File Protection

The UNIX operating system controls access to files with a 16-bit user identification (UID) and a 16-bit group identification (GID).

UNIX file permissions are read, write (including delete), and execute. As with VMS directory files, UNIX systems use the execute access bit to control the search access to a directory. Executable files can have the additional access attributes `setuid` and `setgid`. When a program with `setuid` and `setgid` is executed, the UID and GID of the executing process are changed to allow the process to behave as part of that user or group.

UNIX users are classified into three categories: owner, group, and other. The owner has a matching UID, the group has a matching GID, and other is any other user.

The user categories in UNIX have the following syntax:

```
rwxrwxrwx
```

9.6 VMS File System Restrictions for the NFS Client

Because of the differences between the VMS and UNIX file systems, the NFS server places restrictions on a user accessing VMS files systems on the server. NFS users should be aware of VMS file name conventions and how VMS file name conventions are handled by the NFS server. Guidelines and restrictions are described in the following sections.

9.6.1 Rules for Creating and Referencing Files

When you use files with the NFS client, the file names must conform to VMS file naming conventions. For more information on VMS file naming conventions, see Section 9.2.3.

The following rules apply to creating and referencing files from an NFS client:

- All file names are sent to the NFS client in lowercase.
- Directories can have the .DIR type and version number removed, if the /TYPELESS_DIR option is specified for the export entry.
- By default, version numbers are stripped when only one version exists. (This function can be superseded by defining the logical UCX$CFS_SHOW_VERSION.)
- The name and type separator, a period (.), is removed when the version number has been stripped and the file type is null (contains no character except for a period). The period remains after the name if /TYPELESS_DIR is enabled and a matching directory is present.
- The real VMS files . (a period) and .. (two periods) must be referred to as .. (two periods) plus a version number. On UNIX systems, a period by itself refers to the current directory, and two periods refer to the parent directory.
- On output from the server, the semicolon (;) that separates the type and version is represented as a period (.).
- On input, either a semicolon or a period is accepted as the type and version separator.
- Directory files must have the type and version .DIR.1, unless the /TYPELESS_DIR option is used. The server does not accept a file name that contains a period (.), for it automatically inserts .DIR at creation. (Generally, the .1 is stripped because the VMS operating system explicitly uses only one version (always 1) of the directory file.)

  If you use the /TYPELESS_DIR option, the system removes the .DIR file type and the version number.
- The directory file appears before a matching file name with a null type. For example, the system sees JANE.DIR;1 before it sees JANE.;5.
- When a file is created, its default ownership comes from its directory. If the directory contains an access control list (ACL) and the ACL specifies +DEFAULT, VMS automatically applies an ACL to the newly created file.
- VMS allows only eight levels of directories; this limit is enforced by RMS. If you are using the NFS server to access a VMS system, you can create directory levels more than eight levels deep, but RMS supports only the first eight levels. This may have an undesirable effect if you try to access lower-level files created by the NFS client.

Table 9–1 provides some examples of how the VMS file and directory names would appear to the NFS client user.

Table 9–1 VMS Files As Seen by an NFS Client

On VMS	On NFS Client
..1	..1
ALEX.MAR;551	alex.mar
BEN.DIR;1	ben.dir
GEETHA.C;2	geetha.c.2
GEETHA.C;1	geetha.c.1
JANE.;5	jane
JOHN.;8	john
EVES.GAMES;32767	eves.games

9.6.2 Unsupported Operations

Although the VMS file system does support links, it does not maintain link counts (NLINKS). Therefore, the DEC TCP/IP file system does not support UNIX-style hard links for VMS files. Because the VMS file system does not support symbolic links, soft link creation is also not supported for VMS file systems. Regular files have NLINKS set to 1, and directories have it set to 2.

9.6.3 Protection Restrictions

The following protection restrictions are placed on the NFS client in accessing VMS files:

- Because of the access control lists (ACLs), a particular user can be denied access to a file when the NFS attributes cannot justify access. This happens because the server has denied access and the UNIX system does not recognize ACLs.
- Although the ACL can grant a particular user access to a file on the server, an NFS client can still deny access to the file. The NFS client uses only file protection mask bits to determine access control.
- CONTROL access is enforced against the file, which means the owner of the file has absolute control over the file, despite the protection code.
- The UNIX X protection bit (VMS E protection bit) allows execute access on regular files and search access on directory files. Because the VMS operating system uses the E bit more often than the UNIX operating system uses the X bit, many files appear executable (shell scripts and images), but may not actually be executable.
- When a file is created, the system protection is copied from the owner's protection field. The UNIX operating system does not support system protection.
- The VMS delete protection (D bit) is not supported by the UNIX operating system. When a file is created, the delete protection is a copy of the write protection (W bit). Thus, a file with RWED protection in VMS becomes one with RWX protection in UNIX.

9.6.4 Miscellaneous Restrictions

The following are miscellaneous restrictions that the NFS client user should know about:

- NFS supports only the owner, group, and world mode bits. Therefore, the VMS SYSTEM field is generated with the owner field.
- Directories and regular files are the only supported file types.
- The revision date of the file is updated whenever data is written to the file, not just when it is accessed for a WRITE operation. As a result, the revision date changes more frequently than it normally would for a VMS user.
- VMS does not maintain the last access date, but uses the last revision instead.
- If you have multiple versions of a file and you do not specify an explicit version number, the NFS server uses the latest version of the file. Some NFS client implementations use a file name cache instead. As a result, if the NFS server provides information about a file (for example, FILE.DAT;3), the client would cache the file name as FILE.DAT;3. If you then created a new version of the file with a higher version number (FILE.DAT;4), the NFS client may still return FILE.DAT;3, as long as the client's cache is still active. In this case, you have to specify an explicit version number in order to refer to files with higher version numbers. By default, version numbers are stripped when only one version exists.
- The `chgrp` operations are different for VMS Container file systems, UNIX Container file systems, and native UNIX NFS file systems. This happens because the DEC NFS server does not maintain a local `/etc/group`. (See Section 9.7.)

  When a `chgrp` command is performed on a file mounted on a remote DEC NFS server, the command may act in two possible ways, depending on what file system is mounted.

  If the file system is a CFS UNIX file system, the DEC NFS UNIX server stores the file's owner as UID and GID in the container file. However, because the DEC NFS server does not maintain a `/etc/group` file, the `chgrp` operation is allowed only if the owner of the file changes the file to the owner's GID. For example, if a user with a UID of 10 and GID of 15 owns a file with a UID of 10 and a GID of 17, the following `chgrp` succeeds:

  ```
  chgrp 15 FILE
  ```

  However, if the owner attempts to change the group to any other GID, the operation will fail because the NFS server does not support the `/etc/group` file and therefore cannot identify that the user belongs to the specified group.

  If the file system is a CFS VMS file system, the VMS file system does not save the file's UID and GID. Therefore, this operation is a null operation. No errors are reported, and the GID is not changed. This occurs because the GID that the client observes with the `ls -g` command is a result of a back translation of the VMS UIC into a UID GID pair through the proxy database.
- All users who can access the DEC TCP/IP file systems must be registered in the proxy database.

If a user who is not registered in the proxy database issues a command, an error message appears. For example, if a nonregistered user entered a `df` command, the following message appears:

```
server xxxx: RPC: Authentication error
df: /xxxx: Not owner
```

- The UNIX last access time and change time are not maintained by the VMS operating system. Instead, VMS uses the revision date for both the access time and the change time.
- Files on the VMS server may appear to be a different size than the same files on a UNIX system. This occurs because on a UNIX system, the file size includes both the indirect blocks and the data blocks. On a VMS system, the file size includes only the data blocks.
- DEC NFS always returns the current time as the timestamp on all VMS directory files (.DIR files).

9.7 DEC TCP/IP UNIX File System

The VMS file server provides the NFS clients with access to a UNIX file system. Files created as part of the UNIX file system conform to the UNIX file system rules, not VMS rules. This UNIX file system ensures that existing UNIX applications will work without change.

Each VMS server can support multiple UNIX file systems. A logical UNIX file system is organized as a tree with a single root node, nonleaf nodes being directory files and leaf nodes of the tree being either directory or regular data files. The logical UNIX file system resides on a Files–11 formatted disk and is represented as a set of Files–11 files.

The file system parameters and directory structure for a logical UNIX file system are contained in a single Files–11 data file called a **container file**. Each UNIX regular file is stored as a separate Files–11 data file using a system-assigned valid Files–11 file name. The directory data files in the container file contain the UNIX file names and a pointer to the corresponding Files–11 data file.

There is a VMS directory for each UNIX directory in the container file, and all files cataloged in a UNIX directory are also cataloged in the corresponding VMS directory. However, the UNIX directory hierarchy is not duplicated in the VMS directory hierarchy. UNIX file systems are created with the CREATE CONTAINER command.

Because each UNIX regular file is represented as a VMS data file, VMS applications can use standard access methods (RMS) to access the same data as the NFS client applications.

10

TELNET Commands

This chapter provides reference information for the TELNET commands you can use in command mode.

The commands have two formats, DCL and standard UNIX format. If your local host is a VMS system, you can use either command format.

Table 10–1 lists the VMS and UNIX format of the commands and briefly describes each command.

Table 10–1 TELNET Commands Summary

DCL-Style	UNIX-Style	Description
CONNECT	`open`	Establishes a virtual circuit connection between the local host and the specified remote host.
DISABLE AUTOFLUSH	`toggle autoflush`	Disables the automatic flushing of output when interrupt characters are sent.
DISABLE AUTOSYNCH	`toggle autosynch`	Disables the automatic sending of interrupt characters in urgent mode.
DISABLE BINARY	`toggle binary`	Disables network transmission of data in binary mode.
DISABLE CRLF	`toggle crlf`	Disables the sending of carriage returns as <CR><LF>.
DISABLE CRMOD	`toggle crmod`	Disables the mapping of received carriage returns.
DISABLE DEBUG	`toggle netdata`	Disables the display of network data flow information in hexadecimal.
DISABLE LOCAL_ CHARS	`toggle localchars`	Determines whether certain control characters are interpreted by the local TELNET client and translated to TELNET Commands.
DISABLE OPTIONS_ VIEW	`toggle options`	Determines whether option negotiations between the client and server are displayed.
DISCONNECT	`close`	Terminates the current TELNET session with the remote host.
ENABLE AUTOFLUSH	`toggle autoflush`	Enables the automatic flushing of output when interrupt characters are sent.
ENABLE AUTOSYNCH	`toggle autosynch`	Enables the automatic sending of interrupt characters in urgent mode.
ENABLE BINARY	`toggle binary`	Enables network transmission of data in binary mode.
ENABLE CRLF	`toggle crlf`	Enables the sending of carriage returns as <CR><LF>.
ENABLE CRMOD	`toggle crmod`	Enables the mapping of received carriage returns.
ENABLE DEBUG	`toggle netdata`	Enables the display of network data flow information in hexadecimal.

(continued on next page)

Table 10–1 (Cont.) TELNET Commands Summary

DCL-Style	UNIX-Style	Description
ENABLE LOCAL_ CHARS	`toggle localchars`	Determines whether certain control characters are interpreted by the local TELNET client or the remote TELNET server.
ENABLE OPTIONS_ VIEW	`toggle options`	Determines whether option negotiations between the client and server are displayed.
EXIT	`quit`	Closes any open sessions and exits from the TELNET utility.
HELP	`help` `?`	Invokes the TELNET help facility.
RESUME	Return	Resumes a TELNET session.
SEND AO	`send ao`	Sends the Abort Output command to the remote TELNET server.
SEND AYT	`send ayt`	Sends the Are You There command to the remote TELNET server.
SEND BRK	`send brk`	Sends the TELNET Break command to the remote TELNET server.
SEND EC	`send ec`	Sends the TELNET Erase Character command to the remote TELNET server.
SEND EL	`send el`	Sends the TELNET Erase Line command to the remote TELNET server.
SEND GA	`send ga`	Sends the TELNET Go Ahead command to the remote TELNET server.
SEND IP	`send ip`	Sends the TELNET Interrupt character to the remote TELNET server.
SEND NOP	`send nop`	Sends the TELNET No Operation command to the remote TELNET server.
SEND SYNCH	`send synch`	Sends the TELNET Synchronize character to the remote TELNET server.
SET ECHO	`set echo`	Sets the echo character to the specified character.
SET ERASE	`set erase`	Sets the erase character to the specified character.
SET ESCAPE	`set escape`	Sets the escape character to the specified character.
SET DEVICE	None	Sets the terminal type.
SET FLUSHOUTPUT	`set flushoutput`	Sets the flush output character to the specified character.
SET INTERRUPT	`set intr`	Sets the interrupt character to the specified character.
SET KILL	`set kill`	Sets the kill character to the specified character.
SET MODE	`mode`	Sets the network data transmission mode.
SHOW PARAMETERS	`display`	Displays the current operating parameters.
SHOW STATUS	`status`	Displays information about the current status of TELNET.
SPAWN	`z`	Suspends the current TELNET session and spawns a subprocess.

CONNECT

Establishes a virtual terminal circuit connection between the local host and the specified remote host.

When the connection is established, the remote host's login prompt appears on the screen.

You can optionally specify the remote communication port number. If you do not specify a port number, TELNET attempts to connect to the TELNET server at the default port.

UNIX Command

open *hostname* [*port*]

Format

CONNECT ["]*hostname*["] [*port*]]

Parameters

hostname

Any valid remote host name. The remote host name can be any of the following:

- The actual host name of the remote host
- The Internet address of the remote host
- An alias for the remote host

If the host name contains lowercase letters, you must enclose the name in quotation marks to preserve the case sensitivity.

port

Any valid communication port number on the remote host. In most cases, you should specify the port only if you want to connect to a nonstandard TELNET server.

Examples

1. `TELNET> CONNECT ARIEL`

   Establishes a virtual terminal connection between your local host and the remote host ARIEL.

2. `TELNET> CONNECT 130.180.5.5`

   Establishes a virtual terminal connection between your local host and the remote host at the Internet address 130.180.5.5.

3. `TELNET> CONNECT "apples" 23`

   Establishes a virtual terminal connection between your local host and the remote host `apples`, using communication port number 23 on the remote host.

DISABLE AUTOFLUSH

Disables the automatic flushing of output when the interrupt character is sent. When an interrupt character is sent and autoflush is disabled, the data in the data buffer continues to be displayed until the buffer is empty.

The default is ENABLE AUTOFLUSH.

UNIX Command

```
toggle autoflush
```

Format

DISABLE AUTOFLUSH

Example

```
TELNET> DISABLE AUTOFLUSH
```

Output is not automatically flushed when you use interrupt characters.

DISABLE AUTOSYNCH

Disables the sending of the TELNET synchronization character and interrupt character in urgent mode. (For more information on the synchronization character, see SEND SYNCH.)

In urgent mode, an interrupt character is sent immediately from the local host and processed immediately on the remote host.

When autosynch is disabled, the interrupt character is sent in sequence with the other characters in the stream and processed on the remote host in the sequence it was received.

This is the default.

UNIX Command

```
toggle autosynch
```

Format

DISABLE AUTOSYNCH

Example

```
TELNET> DISABLE AUTOSYNCH
```

Interrupt characters are not sent automatically in urgent mode.

DISABLE BINARY

Disables the network transmission of data in binary mode. When binary mode is disabled, the end-of-line character is always sent as the carriage return/line feed combination (<CR><LF>).

You would use this command when the system you are communicating with expects each line to end with a carriage return/line feed combination.

This is the default.

UNIX Command

```
toggle binary
```

Format

DISABLE BINARY

Example

```
TELNET> DISABLE BINARY
```

The end-of-line character is sent as carriage return/line feed combination (<CR><LF>).

DISABLE CRLF

Disables the sending of carriage returns as the carriage return/line feed combination (<CR><LF>) at the end of each line. When CRLF is disabled, the TELNET client does not ensure that each line ends with <CR><LF>.

You would use this command when the remote system you are connected to expects <CR><NULL> at the end of each line.

The default is ENABLE CRLF.

UNIX Command

```
toggle crlf
```

Format

DISABLE CRLF

Example

```
TELNET> DISABLE CRLF
```

Carriage returns are not sent as a carriage return/line feed combination (<CR><LF>).

DISABLE CRMOD

Disables the mapping of received carriage returns. When CRMOD is disabled, the TELNET client does not ensure that each received line ends with a carriage return/line feed combination (<CR><LF>).

You use this command when your terminal expects <CR><NULL> at the end of each line.

This is the default.

UNIX Command

```
toggle crmod
```

Format

DISABLE CRMOD

Example

```
TELNET> DISABLE CRMOD
```

When you receive carriage returns, they are not mapped.

DISABLE DEBUG

Disables the display of hexadecimal information on network data flow. This mode is suitable for most network communication.

This is the default.

UNIX Command

```
toggle netdata
```

Format

DISABLE DEBUG

Example

```
TELNET> DISABLE DEBUG
```

Hexadecimal information on network data flow is not displayed.

DISABLE LOCAL_CHARS

Disables interpretation of the following VMS terminal control characters by the local TELNET client:

- Ctrl/C — Interrupt
- Ctrl/O — Flush output
- Ctrl/T — Are You There
- Ctrl/U — Kill
- Ctrl/Y — Interrupt

When you DISABLE LOCAL_CHARS, these control characters are sent to the remote server uninterrupted and interpreted by the remote TELNET server. All other terminal control characters are interpreted by the local host. You must be certain that the remote and local host use the same control characters to ensure that the remote host responds appropriately.

DISABLE LOCAL_CHARS is the default.

UNIX Command

```
toggle local_chars
```

Format

DISABLE LOCAL_CHARS

Example

```
TELNET> DISABLE LOCAL_CHARS

TELNET> SHOW PARAMETERS
Will flush output when sending interrupt characters
Won't send interrupt characters in urgent mode
Will map carriage return on output
Won't recognize certain control characters
Won't show option negotiation
Won't print network data flow in hexadecimal
[^E]     echo.
[^]]     escape.
[^?]     erase.
[^O]     flushoutput.
[^C]     interrupt.
[^U]     kill.
```

Certain control characters are not interpreted by the local TELNET client. They are interpreted by the remote TELNET server instead.

DISABLE OPTIONS_VIEW

Disables the display of option negotiations between the local TELNET client and the remote TELNET server during the session. This mode is suitable for most communications.

This is the default.

UNIX Command

```
toggle options
```

Format

DISABLE OPTIONS_VIEW

Example

```
TELNET> DISABLE OPTIONS_VIEW
```

The options negotiations between the client and server are not displayed.

DISCONNECT

Terminates the current TELNET session with the remote server. If you began your session in command mode, you return to command mode. If you began your session in input mode, you exit from TELNET.

If you use the DISCONNECT command to terminate a session on a remote VMS host that has enabled virtual terminals, the connection is closed by the local host; however, the process on the remote host is not terminated. If you want the process to be terminated, you must issue a LOGOUT command.

You must establish a connection before you issue this command.

UNIX Command

```
close
```

Format

DISCONNECT

Examples

1.
```
TELNET> DISCONNECT
Local connection closed
TELNET>
```

   Terminates the current TELNET session and returns you to command mode, because the session was initiated in command mode.

2.
```
TELNET> DISCONNECT
Local connection closed
$
```

   Terminates the current TELNET session and returns you to the system prompt, because the session was initiated in input mode.

ENABLE AUTOFLUSH

Enables the automatic flushing of output when the interrupt character is sent. When an interrupt character is sent and autoflush is enabled, the data buffer is emptied and the display is terminated.

You use this command when you want fast system response to the interrupt character.

This is the default.

UNIX Command

```
toggle autoflush
```

Format

ENABLE AUTOFLUSH

Example

```
TELNET> ENABLE AUTOFLUSH
```

Output is automatically flushed when you send the interrupt character.

ENABLE AUTOSYNCH

Enables the sending of the TELNET synchronization character and interrupt character in urgent mode. (For more information on the synchronization character, see SEND SYNCH.)

In urgent mode, an interrupt character is sent immediately from the local host and processed immediately on the remote host.

You would use this command when you want faster response time to the TELNET synchronization character and interrupt character.

The default is DISABLE AUTOSYNCH.

UNIX Command

```
toggle autosynch
```

Format

ENABLE AUTOSYNCH

Example

```
TELNET> ENABLE AUTOSYNCH
```

The interrupt character is sent automatically in urgent mode.

ENABLE BINARY

Sets the network transmission of data to binary mode. When binary mode is enabled, the end-of-line character is not mapped to the carriage return/line feed combination (<CR><LF>).

You use binary mode when you want to interact with processes on the remote host that expect special characters to be input into them, but the remote host does not support automatic negotiation of the TELNET binary option.

The default is DISABLE BINARY.

UNIX Command

```
toggle binary
```

Format

ENABLE BINARY

Example

```
TELNET> ENABLE BINARY
```

Sets the network transmission of data to binary mode. The end-of-line character is not mapped to the carriage return/line feed combination (<CR><LF>).

ENABLE CRLF

Enables the sending of carriage returns as the carriage return/line feed combination (<CR><LF>) at the end of each line. When CRLF is enabled, the TELNET client ensures that each line ends with a <CR><LF>.

You use this command when the system you are connected to expects <CR><LF> at the end of each line.

This is the default.

UNIX Command

```
toggle crlf
```

Format

ENABLE CRLF

Example

```
TELNET> ENABLE CRLF
```

Carriage returns are sent as <CR><LF>.

ENABLE CRMOD

Enables the mapping of received carriage returns. When CRMOD is enabled, the TELNET client ensures that each line you receive ends with the carriage return/line feed combination (<CR><LF>).

You use this command when the system you are communicating with expects each line to end with <CR><LF>.

The default is DISABLE CRMOD.

UNIX Command

```
toggle crmod
```

Format

ENABLE CRMOD

Example

```
TELNET> ENABLE CRMOD
```

When you receive carriage returns, they are mapped to the carriage return/line feed combination (<CR><LF>).

ENABLE DEBUG

Enables the display of network data flow information in hexadecimal. Use debug when you have problems during a session. Enabling debug mode can help you determine where the problem is occurring and what parameters to use during the session.

The default is DISABLE DEBUG.

UNIX Command

```
toggle netdata
```

Format

ENABLE DEBUG

Example

```
TELNET> ENABLE DEBUG

TELNET> CONNECT ARIEL
Trying...
Connected to ARIEL.
Escape character is '^]'.

> 0x0   fffd1fffd3fffb18
< 0x0   fffd18

< 0x0   fffb1fffb3fffa181fff0

> 0x0   fffa1805654333030fff0
< 0x0   fffd1dada556c747269782d33322056332e3020285265762036342920 28
< 0x20  726176696e652e7a6b332e6465632e636f6d29daddad

UNIX V4.0 (Rev. 9) (ARIEL)

> 0x0   fffc1
< 0x0   fffe1

< 0x0   6c6f67696e3a20
login:
```

Hexadecimal network data flow information is displayed.

ENABLE LOCAL_CHARS

Enables the interpretation of the following VMS terminal control characters by the local TELNET client:

- Ctrl/C — Interrupt
- Ctrl/O — Flush output
- Ctrl/T — Are You There
- Ctrl/U — Kill
- Ctrl/Y — Interrupt

When you ENABLE LOCAL_CHARS, the control characters are interpreted by the local TELNET client and translated to TELNET commands. For example, Ctrl/C is translated and sent to the remote TELNET server as the IP TELNET command.

You use this command when the remote host and your local host have different control characters. By having your local host interpret the control characters before sending them to the remote host, you can be certain that the remote host responds appropriately.

The default is DISABLE LOCAL_CHARS.

UNIX Command

```
toggle localchars
```

Format

ENABLE LOCAL_CHARS

Example

```
TELNET> ENABLE LOCAL_CHARS

TELNET> SHOW PARAMETERS
Will flush output when sending interrupt characters
Won't send interrupt characters in urgent mode
Will map carriage return on output
Will recognize certain control characters
Won't show option negotiation
Won't print network data flow in hexadecimal
[^E]     echo.
[^]]     escape.
[^?]     erase.
[^O]     flushoutput.
[^C]     interrupt.
[^U]     kill.
```

Control characters are interpreted by the local TELNET client instead of the remote TELNET server.

ENABLE OPTIONS_VIEW

Enables the display of option negotiations between the local TELNET client and the remote TELNET server during the session. You use this command when you have problems during a session. Displaying the options negotiations can help you determine where the problem is occurring and what parameters to use during the session.

The default is DISABLE OPTIONS_VIEW.

UNIX Command

```
toggle options
```

Format

ENABLE OPTIONS_VIEW

Example

```
TELNET> ENABLE OPTIONS_VIEW

TELNET> CONNECT ARIEL
SENT do ECHO
SENT do SUPPRESS GO AHEAD
SENT will TERMINAL TYPE (don't reply)
Trying...
Connected to ARIEL.
Escape character is '^]'.

RCVD do TERMINAL TYPE (don't reply)
RCVD will ECHO (don't reply)
RCVD will SUPPRESS GO AHEAD (don't reply)
Received suboption Terminal Type - request to send.
Terminal type sent-VT300
SENT wont ECHO (don't reply)

UNIX-32 V3.0 (Rev 64) (ARIEL)


RVCD don't ECHO (don't reply)
login: userb
Password:
```

The options negotiations between the client and server are displayed.

EXIT

Closes any open sessions and exits from the TELNET utility. Alternatively, you can use Ctrl/Z.

If you use the EXIT command to terminate a session on a remote VMS host that has enabled virtual terminals, the connection is closed by the local host; however, the process on the remote host is not terminated. If you want the process to be terminated, you must issue a LOGOUT command.

UNIX Command

```
quit
```

Format

EXIT

Example

```
TELNET> EXIT
$
```

Closes any open TELNET session and exits from TELNET.

HELP

Invokes the TELNET help facility, which provides information on using TELNET and its commands.

If you do not specify a keyword with the HELP command, it displays an overview of TELNET and a list of additional help topics.

If you do specify a keyword, it displays information on that keyword.

Equivalent UNIX Commands

```
help
```

```
?
```

Format

HELP [*keyword*]

Parameter

[keyword]
Specifies a keyword on which you want information. VMS wildcards are permitted.

Examples

1. `TELNET> HELP`

   Provides an overview of TELNET and a list of additional help topics.

2. `TELNET> HELP CONNECT`

   Displays information about the CONNECT command.

RESUME

Resumes the current TELNET session.

You use this command after you have entered an escape command and wish to resume the TELNET session.

You must establish a connection before using this command.

UNIX Command

Return

Format

RESUME

Example

```
ARIEL> Ctrl/]
TELNET> RESUME
Session Resumed
```

In this example the following sequence occurs:

1. From the remote system prompt, the TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
2. Control is returned to the local TELNET client.
3. The RESUME command is entered.
4. The session is resumed.

SEND AO

Aborts the output of the last command you entered on the remote host; however, the command itself continues to execute. If you issue another SEND AO (Abort Output) command, the output resumes, provided the original command has not finished execution.

You use this command when you are communicating with a system that does not recognize Ctrl/O as the flush output character. You can also use this command when you want to terminate the output, but not the execution of the process.

You must establish a connection before using this command.

UNIX Command

send ao

Format

SEND AO

Example

```
remote> cd /bin
remote> ls -1 -l
total 1464
-rwxr-xr-x  2 root        32768 Oct 19  1988 STTY
-rwxr-xr-x  2 root         5120 Oct 19  1988 [
-rwxr-xr-x  1 root        45056 Oct 19  1988 adb
lrwxr-xr-x  1 root           13 Aug 21 17:41 ar -> ../usr/bin/ar
Ctrl/]
TELNET> SEND AO
```

In this example, the following sequence occurs:

1. The directory on the remote host is set to /bin.
2. An ls command is entered.
3. The TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
4. The SEND AO command is entered.
5. The output of the ls command is aborted.

SEND AYT

Determines whether you are still connected to the remote host. (AYT stands for Are You There.)

You must establish a connection before using this command.

UNIX Command

```
send ayt
```

Format

SEND AYT

Examples

1.
```
ARIEL> Ctrl/]
TELNET> SEND AYT

[Yes]
```

   Determines whether you are still connected to the remote host. The [Yes] response indicates that you are still connected to the UNIX system.

2.
```
TELNET> SEND AYT

 DEC TCP/IP for VMS 2.0 VMS TELNET Server

DOGGIE::USERNAME 10:54:24 UCX$TELNE CPU=00:00:45.34 PF=5419 IO=4647
MEM=456
TELNET>
```

   Shows that you are still connected to a remote VMS system.

SEND BRK

Terminates execution of the last command you entered on the remote host.

You use this command when you are communicating with a host that does not recognize Ctrl/C as an interrupt character. (BRK stands for Break.)

You must establish a connection before using this command.

UNIX Command

send brk

Format

SEND BRK

Example

```
remote> cd /bin
remote> ls -1 -l
total 1464
-rwxr-xr-x  2 root        32768 Oct 19  1988 STTY
-rwxr-xr-x  2 root         5120 Oct 19  1988 [
-rwxr-xr-x  1 root        45056 Oct 19  1988 adb
lrwxr-xr-x  1 root           13 Aug 21 17:41 ar -> ../usr/bin/ar
lrwxr-xr-x  1 root           13 Aug 21 17:41 as -> ../usr/bin/as
Ctrl/]
TELNET> SEND BRK
```

In this example, the following sequence occurs:

1. The directory on the remote host is set to /bin.
2. An ls command is entered.
3. The TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
4. The SEND BRK command is entered.
5. The execution of the ls command is terminated.

SEND EC

Erases the last character typed on the remote host. You use this command when the remote host you are connected to does not recognize your delete key. (EC stands for Erase Character.)

You must establish a connection before using this command.

UNIX Command

`send ec`

Format

SEND EC

Example

```
ARIEL> mailt Ctrl/]
TELNET> SEND EC Return
Mail version 2.18 5/19/83.  Type ? for help.
"/usr/spool/mail/smith": 1 message 1 new
>N  1 smith  Mon Aug 28 13:39  8/161  "FYI: Q4 Earnings"
&
```

In this example, the following sequence occurs:

1. The UNIX `mail` command is misspelled.
2. The TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
3. TELNET returns to command mode.
4. The SEND EC command is entered, causing the last character typed on the remote host (`t`) to be erased.
5. Control is returned to the remote host.
6. Return is pressed and the `mail` command is executed.

SEND EL

Erases the last line of text entered on the remote host. You use this command when the remote host you are connected to does not recognize your delete key or does not respond to command line recall. (EL stands for Erase Line.)

You must establish a connection before using this command.

UNIX Command

```
send el
```

Format

SEND EL

Example

```
ARIEL> amil Ctrl/]
TELNET> SEND EL Return
mail Return
Mail version 2.18 5/19/83.  Type ? for help.
"/usr/spool/mail/smith": 1 message 1 new
>N  1 smith  Mon Aug 28 13:39  8/161  "FYI: Q4 Earnings"
&
```

In this example, the following sequence occurs:

1. The UNIX `mail` command is misspelled.
2. The TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
3. TELNET returns to command mode.
4. The SEND EL command is entered, which causes the following events to occur:
   - The line (amil) is erased.
   - Control is returned to the remote host.
5. The `mail` command is reentered and executed.

SEND GA

Signals the remote host that the client is ready to receive requests. (GA stands for Go Ahead.)

During most TELNET sessions, the Go Ahead option is suppressed and this command has no effect. There are, however, some applications (such as terminal-to-terminal communications) that may require GA commands to be sent in either one or both directions.

This command is provided for those applications and implementations.

You must establish a connection before using this command.

UNIX Command

```
send ga
```

Format

SEND GA

Example

```
ARIEL> Ctrl/]
TELNET> SEND GA
```

Sends the TELNET Go Ahead command to the remote host.

SEND IP

Terminates the execution of the last command you entered on the remote host. You use this command when you are communicating with a host that does not recognize Ctrl/C as an interrupt character.

You must establish a connection before using this command.

UNIX Command

```
send ip
```

Format

SEND IP

Example

```
remote> cd /bin
remote> ls -1 -l
total 1464
-rwxr-xr-x  2 root        32768 Oct 19  1988 STTY
-rwxr-xr-x  2 root         5120 Oct 19  1988 [
-rwxr-xr-x  1 root        45056 Oct 19  1988 adb
lrwxr-xr-x  1 root           13 Aug 21 17:41 ar -> ../usr/bin/ar
lrwxr-xr-x  1 root           13 Aug 21 17:41 as -> ../usr/bin/as
Ctrl/]
TELNET> SEND IP
```

In this example, the following sequence occurs:

1. The directory on the remote host is set to `/bin`.
2. An `ls` command is entered.
3. The TELNET escape sequence is entered. (The escape character is not echoed to the screen.)
4. The SEND IP command is entered.
5. The execution of the `ls` command is terminated.

SEND NOP

Determines if you are connected or if your connection is still alive. (NOP stands for No Operation.)

If you receive an error message when you send the NOP command, it indicates that there is a problem with the connection. If you do not receive an error message, it means the connection is active.

You must establish a connection before using this command.

UNIX Command

```
send nop
```

Format

SEND NOP

Examples

1. % Ctrl/]
   TELNET> SEND NOP
   RETURN
   TELNET>

   In this example, the following sequence occurs:

   - The escape key is pressed.
   - The NOP command is sent to the remote host.
   - The Return key is pressed to resume the current session.

   Because no error message is received, the connection is still alive.

2. $ TELNET

   TELNET> SEND NOP
   Need to be connected first
   TELNET>

   In this example, the following sequence occurs:

   - You enter into TELNET command mode.
   - The NOP command is sent.
   - An error message is received that indicates that you must be connected to a remote host to use the SEND NOP command.

3. % Ctrl/J
```
TELNET> SEND NOP
Failed to write to a socket
TELNET>
```

In this example, the following sequence occurs:

- The escape key is pressed.
- The NOP command is entered.
- The error message received indicates that the connection has been broken.

SEND SYNCH

Clears the communication path between the TELNET client and server by sending the TELNET synchronization character to the remote host. When the TELNET synchronization character is sent to the remote host, the server discards the incoming data stream and flushes all unreceived data.

You use this command when you want the remote host to disregard any incoming data it hasn't processed.

You must establish a connection before using this command.

UNIX Command

```
send synch
```

Format

SEND SYNCH

Example

```
TELNET> SEND SYNCH
```

Sends the TELNET Synchronize character to the remote host.

SET DEVICE

Sets the terminal type.

Once a connection is established, the TELNET client and server perform session negotiations. Part of the negotiations is determining the terminal type. To set the terminal type, the client sends a message to the server identifying the type of terminal you are working from. Usually, this is recognized by the remote server. If the terminal type is not recognized by the remote server, the terminal type is set to unknown.

You use the SET DEVICE command in cases like these, where the remote host does not recognize the terminal type. In this situation, you can set the terminal type to VT100, which is widely recognized as a standard terminal type. However, you can attempt to set the terminal type to any standard Digital terminal type.

UNIX Command

None.

Format

SET DEVICE

Command Qualifier	Defaults
/TERMINAL=*terminal_type*	See text.

Command Qualifier

/TERMINAL=terminal_type

Specifies the terminal type to the remote host.

The terminal type defaults to that of the terminal you are using.

Example

```
TELNET> SET DEVICE/TERMINAL=VT100
```

Sets the terminal type to VT100.

SET ECHO

Sets the echo character in a specified character control sequence.

To set the echo character to a control character, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the echo character control sequence to Ctrl/N:

```
TELNET> SET ECHO "^N"
```

You use the SET ECHO command when your terminal or the remote system you are connected to does not recognize the default echo character control sequence, Ctrl/E.

UNIX Command

set echo *echo_char*

Format

SET ECHO echo_char

Parameter

echo_char
Any character.

Example

```
TELNET> SET ECHO "^m"
Echo character is '^M'.
```

Sets the echo character control sequence to Ctrl/M.

SET ERASE

Sets the erase character in a specified character control sequence.

When you type the erase character, it causes the last character in the type-ahead buffer to be erased, either locally or by sending the remote host the EC sequence. The erase character has no effect in binary mode.

To set the erase character in a control sequence, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the erase character control sequence to Ctrl/N:

```
TELNET> SET ERASE "^N"
```

You use the SET ERASE command when your terminal or the remote system you are connected to does not recognize the default erase character sent by pressing the delete key.

UNIX Command

`set erase` *erase_char*

Format

SET ERASE erase_char

Parameter

erase_char
Any character.

Example

```
TELNET> SET ERASE "^p"
 character is '^p'.
```

Sets the erase character control sequence to Ctrl/P.

SET ESCAPE

Sets the TELNET escape character in a specified character control sequence.

To set the escape character in a character control sequence, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the escape character control sequence to Ctrl/N:

```
TELNET> SET ESCAPE "^N"
```

You use the SET ESCAPE command when your terminal or the remote system you are connected to does not recognize the default escape character control sequence, Ctrl/].

When you press the escape key, you return to the TELNET command level on the host where you initially logged in. Therefore, if you have have created a TELNET session with another system during your original TELNET session, you would return to the initial host. You can use the SET ESCAPE command to have different escape sequences to return to TELNET command level for the different connections you have established.

UNIX Command

`set escape` *escape_char*

Format

SET ESCAPE escape_char

Parameter

escape_char
Any character.

Example

```
TELNET> SET ESCAPE "^z"
 character is '^Z'.
```

Sets the escape character control sequence to Ctrl/Z.

SET FLUSHOUTPUT

Sets the flush output character in the specified character control sequence.

To set the flush output character in a character control sequence, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the flush output character control sequence to Ctrl/N:

```
TELNET> SET FLUSHOUTPUT "^N"
```

You use the SET FLUSHOUTPUT command when your terminal or the remote system you are connected to does not recognize the default flush output character, Ctrl/O.

UNIX Command

`set flushoutput` *flush_char*

Format

SET FLUSHOUTPUT flush_char

Parameter

flush_char
Any character.

Example

```
TELNET> SET FLUSHOUTPUT "^v"
 character is '^V'.
```

Sets the flush output character control sequence to Ctrl/V.

SET INTERRUPT

Sets the interrupt character in a specified character control sequence.

The interrupt character causes the interrupt (IP) sequence to be sent in urgent mode. This clears the input and output paths to the remote host. The remote host interprets the IP sequence to mean that you want to interrupt the program that is processing.

The interrupt character has no effect in binary mode.

To set the interrupt character in a character control sequence, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the interrupt character control sequence to Ctrl/N:

```
TELNET> SET INTERRUPT "^N"
```

You use the SET INTERRUPT command when your terminal or the remote system you are connected to does not recognize the default interrupt character control sequence, Ctrl/C.

UNIX Command

`set intr` *interrupt_char*

Format

SET INTERRUPT interrupt_char

Parameter

interrupt_char
Any character.

Example

```
TELNET> SET INTERRUPT "^a"
 character is '^A'.
```

Sets the interrupt character control sequence to Ctrl/A.

SET KILL

Sets the kill character in a specified character control sequence.

The kill character causes the entire type-ahead buffer to be discarded, both locally and remotely, by sending the erase line (EL) sequence to the remote host.

The kill character has no effect in binary mode.

To set the kill character in a character control sequence, you must precede the character with a circumflex (^) and place both the circumflex and the character in quotation marks. For example, the following procedure sets the kill character control sequence to Ctrl/N:

```
TELNET> SET KILL "^N"
```

You use the SET KILL command when your terminal or the remote system you are connected to does not recognize the default kill character control sequence, Ctrl/U.

UNIX Command

`set kill` *kill_char*

Format

SET KILL kill_char

Parameter

kill_char
Any character.

Example

```
TELNET> SET KILL "^q"
 character is '^Q'.
```

Sets the kill character control sequence to Ctrl/Q.

SET MODE

Sets the network data transmission mode.

The local host asks the remote host for permission to go into the requested mode. The remote host enters the requested mode if it can.

You can set the mode to either CHAR (for character mode) or LINE (for line mode). In character mode, data is transmitted a character at a time. In line mode, data is transmitted a line at a time.

You must establish a connection before you can change the mode.

Character mode is the default.

UNIX Command

mode *mode_name*

Format

SET MODE mode_name

Restriction

Text editors that do character mode processing may not work properly in TELNET line mode. Therefore, if you use an editor that does character processing, use TELNET in character mode.

Parameter

mode_name
Either CHAR or LINE.

CHAR establishes character mode. In character mode, data is transmitted a character at a time.

LINE establishes line mode. In line mode, data is transmitted a line at a time.

Examples

1. TELNET> SET MODE CHAR

   Sets the transmission of data to character mode.

2. TELNET> SET MODE LINE

   Sets the transmission of data to line mode.

SHOW PARAMETERS

Displays the current operating parameters.

You use this command to determine the parameters for the current TELNET session.

UNIX Command

display

Format

SHOW PARAMETERS

Example

```
TELNET> SHOW PARAMETERS
Will flush output when sending interrupt characters
Won't send interrupt characters in urgent mode
Will map carriage return on output
Won't recognize certain control characters
Won't show option negotiation
Won't print network data flow in hexadecimal
[^E]     echo.
[^]]     escape.
[^?]     erase.
[^O]     flushoutput.
[^C]     interrupt.
[^U]     kill.
```

Displays the operating parameters.

SHOW STATUS

Displays information about the current TELNET session.

You use this command to determine the status for the current TELNET session.

UNIX Command

```
status
```

Format

SHOW STATUS

Example

```
TELNET> SHOW STATUS
Connected to remote
Operating in Character-at-a-time mode.
Escape character is '^]'.
Options:
        Echo -- Remote
        Suppress Go Ahead
        Terminal Type -- VT300

Terminal Data overruns: 0
Suspended Network I/Os: 0
```

Displays information about the current TELNET session.

SPAWN

Suspends the current TELNET session and spawns a subprocess running the DCL command interpreter on the local VMS host.

You use this command when you want to return to DCL command level on the client host.

UNIX Command

```
Z
```

Format

SPAWN

Example

```
TELNET> SPAWN
$
```

Spawns a subprocess on the local VMS host.

11

File Transfer Protocol Commands

This chapter provides reference information for the File Transfer Protocol (FTP) commands.

Table 11–1 lists the DCL-style and standard (UNIX) FTP format of the commands and briefly describes each command.

Table 11–1 FTP Commands Summary

DCL-Style	UNIX-Style	Description
APPEND	`append`	Appends the local file to the specified file on the remote host.
CONNECT	`open`	Establishes a connection to the specified FTP server.
CREATE/DIRECTORY	`mkdir`	Creates the specified directory on the remote host.
DELETE	`delete` `mdelete`	Deletes one or more files on the remote host.
DIRECTORY	`ls`	Lists the files in the specified directories.
DISABLE LOG	`debug`	Disables the display of commands that are sent to the remote server.
DISABLE PORT_COMMAND	`sendport`	Disables the sending of the PORT command.
DISABLE PARSE	`glob`	Disables the expansion of file name specifications.
DISABLE REPLY	`verbose`	Disables the display of all responses from the remote server.
DISABLE TRANSFER_ VERIFICATION	`hash`	Disables the display of a number sign for each data buffer transferred.
DISCONNECT	`close` `disconnect`	Terminates the FTP session with the remote sever.
ENABLE LOG	`debug`	Enables the display of commands that are sent to the remote server.
ENABLE PARSE	`glob`	Enables the expansion of file name specifications.
ENABLE PORT_COMMAND	`sendport`	Enables the sending of the port command
ENABLE REPLY	`verbose`	Enables the display of all responses from the remote server.
ENABLE TRANSFER_ VERIFICATION	`hash`	Enables the display of a number sign for each data buffer transferred.
EXIT	`quit`	Closes any open connection with the remote server and exits FTP.
GET	`get` `mget`	Copies one or more files from the remote host to the local host.

(continued on next page)

Table 11–1 (Cont.) FTP Commands Summary

DCL-Style	UNIX-Style	Description
HELP	help ?	Invokes the local FTP help facility.
HELP/REMOTE	remotehelp	Invokes the remote FTP help facility.
LOGIN	user	Lets you log in to an account on the remote server.
PUT	put mput	Copies one or more files from the local host to the remote host.
QUOTE	quote	Sends the arguments directly to the remote FTP server.
RENAME	rename	Renames the specified file on the remote host.
SET DEFAULT	cd lcd	Sets the working directory to the specified directory on the remote host or on the local host if the /LOCAL qualifier is used.
SET MODE	mode	Defines how data is to be transferred.
SET STRU	struct	Defines the structure of a file to be transferred.
SET TYPE	type	Defines the data representation for the data transfer.
SHOW DEFAULT	pwd	Displays the name of the current working directory.
SHOW STATUS	status	Displays the current status of FTP.
SPAWN	!	Spawns a subprocess running the DCL command interpreter on the local VMS host.

APPEND

Appends the local file to the specified file on the remote host.

UNIX Command

append *local_file remote_file*

Format

APPEND *local_file* ["] *remote_file*["]

Restriction

Wildcards are not permitted.

Parameters

local_file

A valid VMS file specification.

remote_file

A valid file specification to the remote host.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

Example

```
FTP> APPEND LOCAL.COM "rmtcom"
---> PORT 130,180,4,8,4,84
200 PORT command successful.
---> APPE RMTCOM
150 Opening data connection for RMTCOM (130.180.4.8,1108).
226 Transfer complete.
2819 bytes sent in 00:00:00.11 seconds
```

Appends the file LOCAL.COM to the file `rmtcom`.

CONNECT

Establishes a connection to the specified FTP server (host).

After a connection is established, FTP automatically prompts for a user name and password so you can log in to the connected remote host.

UNIX Command

open *host*

Format

CONNECT ["]*hostname*["]

Parameter

hostname

The name or Internet address of the remote host to which you want to establish a connection from your local host.

To preserve case sensitivity, place lowercase host names in quotation marks.

Examples

1.
```
FTP> CONNECT "june"
220 june.vk.com FTP server (ULTRIX Version 4.0 Wed Jun 2 23:28:14 EDT
1989) ready.
Connected to june.vk.com.
Name (june:smith):
```

   Establishes a connection with a remote ULTRIX host called june and prompts you for a user name.

2.
```
FTP> CONNECT MAITAI
220 maitai FTP server ready.
Connected to MAITAI.
Name (maitai:smith):
```

   Establishes a connection with a remote VMS host called MAITAI and prompts you for a user name.

3.
```
FTP> CONNECT 130.180.4.9
220 moon.vk.com FTP server (ULTRIX Version 4.0 Wed Jun 3 13:24:14 EDT
1989) ready.
Connected to moon.vk.com.
Name (130.180.4.9:smith):
```

   Establishes a connection with the remote host with the Internet address 130.180.4.9 and prompts you for a user name.

CREATE/DIRECTORY

Creates the specified directory on the remote host.

UNIX Command

`mkdir` *directory_name*

Format

CREATE/DIRECTORY ["]*directory_name*["]

Parameter

directory_name

The name of the directory to be created on the remote host.

Any directory specification that does not conform to VMS directory specification rules must be placed in quotation marks.

Examples

1.
```
FTP> CREATE /DIRECTORY "temp"
257 MKD command successful.
```

   If you are connected to a UNIX system and your working directory is `/usr/staff/dir`, then this command creates the directory `/usr/staff/dir/temp`.

2.
```
FTP> CREATE /DIRECTORY [.TEMP]
200 MKD command okay.
```

   If you are connected to a VMS system and your working directory is DUA0:[DIR], then this command creates the directory DUA0:[DIR.TEMP].

DELETE

Deletes one or more files on the remote host.

UNIX Command

`delete` *remote_file*[, ...]

`mdelete` *remote_file* [...]

Format

DELETE ["]*remote_file*["], ...

Command Qualifier	Default
/DIRECTORY	None.

Parameter

remote_file

The name of the file to be deleted on the remote host. You can use a wildcard in the file specification. A VMS file requires a version number to be specified.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

You can specify multiple files by separating them with commas (,).

Command Qualifier

/DIRECTORY

Specifies that the remote file to be deleted is a directory file.

Examples

1.
```
FTP> DELETE "myfile"
250 DELE command successful.
```

   Deletes the file `myfile` on the remote UNIX host.

2.
```
FTP> DELETE MYFILE.TXT;2
200 DELETE command okay.
```

   Deletes the file MYFILE.TXT;2 on the remote VMS host.

3.
```
FTP> DELETE/DIRECTORY MYDIR.DIR;1
```

   Deletes the directory MYDIR.DIR;1 on the remote VMS host.

DIRECTORY

Lists the files in the specified directory or directories. By default, the brief form of listing is provided.

You can also use the DIRECTORY command to obtain directory information from DECnet hosts if your FTP client is connected to the DECnet host through a host running DEC TCP/IP software. In this case, you must use the full file specification, including the host, device, directory, and file name.

UNIX Command

`ls` *directory_name*[, ...]

Format

DIRECTORY [["]*directory_name*["][,...]

Command Qualifiers	Defaults
/FULL	None.
/OUTPUT=file_name	None.

Parameter

directory_name

The name of the directory whose files are to be listed. If you do not use this parameter, the default directory is listed.

You can use a wildcard in the directory name.

You can specify multiple files by separating them with either a comma (,) or a plus sign (+).

Any directory specification that does not conform to VMS directory specification rules must be placed in quotation marks.

Command Qualifiers

/FULL

Provides a long-form display of information about files in the directory.

/OUTPUT=file_name

Directs the output of the DIRECTORY command to the specified file.

Examples

1.
```
FTP> DIRECTORY "/usr/temp"
---> PORT 130,180,9,8,9,126
200 PORT command successful.
---> NLST
150 Opening data connection for /bin/ls (130.180.9.8,1150) (0 bytes).
.emacs_1268
.x.txt
x.txt
z.txt

226 Transfer complete.
35 bytes received in 00:00:00.02 seconds
```

Displays the list of files found in the UNIX directory `/usr/temp`.

2.
```
FTP> DIRECTORY [USER]
---> PORT 130,180,4,10,4,134
200 PORT command okay.
---> NLST
150 Opening data connection for WORK3$:[USER]*.*.* (130.180.4.10,1158)
A.TXT;1
TM.TXT;1
X.TXT;1

226 Transfer complete.
28 bytes received in 0.4.80:00:00.01 seconds
```

   Displays the list of files found in the VMS directory USER.

3.
```
FTP> DIRECTORY ARIEL::DEVICE:[USER]MEMO.TXT
```

   Uses the DIRECTORY command to get information from a host connected to the client through a host running DEC TCP/IP software.

DISABLE LOG

Turns off the display of commands that are sent to the remote server. This option is disabled by default. When the LOG option is enabled, the commands that are sent to the remote server are preceded by an arrow (—>).

UNIX Command

verbose

Format

DISABLE LOG

Example

```
FTP> DISABLE LOG
FTP> PUT X.TXT
200 PORT command successful.
150 Opening data connection for X.TXT (130.180.4.10,1165).
226 Transfer complete.
3 bytes sent in 00:00:00.00 seconds
```

Turns off the display of commands that are sent to the remote server. When the LOG option is enabled, the commands that are sent to the remote server are indicated by an arrow (—>).

DISABLE PARSE

Disables the expansion of file name specifications during GET and PUT operations. When parsing is disabled, the file name arguments are taken literally and not expanded.

During PUT operations, the expansion is done by the local host.

UNIX Command

```
glob
```

Format

DISABLE PARSE

Examples

1.
```
FTP> DISABLE PARSE
FTP> PUT X*.TXT
%UCX-E-FTP_WLDCRD, Wildcard is not allowed for REMOTE file
```

   Because parsing is disabled, wildcards are not allowed for the PUT command.

2.
```
FTP> DISABLE PARSE
FTP> GET X.*
200 PORT command successful.
%UCX-E-FTP_OPNOUT, Error opening for output : X.*
-RMS-F-WLD, invalid wildcard operation.
```

   Because parsing is disabled, wildcards are not allowed for the GET command; therefore, the file specification (X.*) is sent as a literal.

DISABLE PORT_COMMAND

Turns off the sending of the PORT command to the remote FTP server.

By default, FTP sends a PORT command when establishing a connection for each data transfer. If the PORT command fails, FTP uses the default data port.

Disable the PORT command when communicating with certain FTP implementations that ignore PORT commands.

UNIX Command

```
sendport
```

Format

DISABLE PORT_COMMAND

Example

```
FTP> DISABLE PORT_COMMAND
```

Turns off the sending of the PORT command.

DISABLE REPLY

Turns off the display of all responses from the remote server. The default is ENABLE REPLY.

UNIX Command

verbose

Format

DISABLE REPLY

Example

```
FTP> DISABLE REPLY
FTP> PUT X.TXT
---> PORT 130,180,4,10,4,152
---> STOR X.TXT
3 bytes sent in 00:00:00.00 seconds
```

Turns off the display of all responses from the remote server.

DISABLE TRANSFER_VERIFICATION

Turns off the display of a number sign (#) for each data buffer transferred over the network. This is the default.

UNIX Command

```
hash
```

Format

DISABLE TRANSFER_VERIFICATION

Example

```
FTP> DISABLE TRANSFER_VERIFICATION
```

Disables the display of a number sign (#) character for each data buffer transferred over the network.

DISCONNECT

Terminates the FTP session with the remote server and returns to the FTP command interpreter.

UNIX Command

```
close
```

```
disconnect
```

Format

DISCONNECT

Example

```
FTP> DISCONNECT
221 Goodbye.
FTP>
```

Terminates the FTP session with the remote server.

ENABLE LOG

Enables each command sent to the remote host to be displayed. This is the default.

UNIX Command

verbose

Format

ENABLE LOG

Example

```
FTP> ENABLE LOG
FTP> ENABLE REPLY
FTP>PUT X.TXT
---> PORT 130,180,4,10,4,154
---> STOR X.TXT
609 bytes sent in 00:00:00.02 seconds
```

Enables each command sent to the remote host to be displayed.

ENABLE PARSE

Enables the expansion of file name specifications during GET and PUT operations. When parsing is enabled, the wildcard characters are recognized. Therefore, file name arguments that contain wildcards are expanded.

During PUT operations, the expansion is done by the local host. During GET operations, the expansion is done on the remote host. Therefore, during GET operations, expansion of a directory name may be different from expansion of other file names. The result depends on the remote host's operating system and the FTP server.

UNIX Command

```
glob
```

Format

ENABLE PARSE

Examples

1.
```
FTP> ENABLE PARSE
FTP> PUT X*.TXT
```

   Because parsing is enabled, wildcards are expanded. Therefore, this PUT command is successful.

2.
```
FTP> ENABLE PARSE
FTP> GET X.*
```

   Because parsing is enabled, wildcards are expanded. Therefore, this GET command is successful.

ENABLE PORT_COMMAND

Enables the sending of the PORT command if it has been disabled.

By default, FTP sends a PORT command when establishing a connection for each data transfer. If the PORT command fails, FTP uses the default data port.

The default is ENABLE PORT_COMMAND.

UNIX Command

sendport

Format

ENABLE PORT_COMMAND

Example

```
FTP> ENABLE PORT_COMMAND
FTP> PUT X.TXT
---> PORT 130,180,4,10,4,158
200 PORT command successful.
---> STOR X.TXT
150 Opening data connection for X.TXT (130.180.10.8,1182).
226 Transfer complete.
3 bytes sent in 00:00:00.01 seconds
```

When the PORT_COMMAND command is enabled, FTP issues a port command for each data transfer.

ENABLE REPLY

Enables the display of all the responses from the FTP server. The default is DISABLE REPLY.

UNIX Command

verbose

Format

ENABLE REPLY

Example

```
FTP> ENABLE REPLY
FTP>PUT X.TXT
---> PORT 130,180,4,10,4,160
200 PORT command successful.
---> STOR X.TXT
150 Opening data connection for X.TXT (130.180.4.10,1184).
226 Transfer complete.
```

Enables all the responses from the FTP server to be displayed.

ENABLE TRANSFER_VERIFICATION

Enables the display of a number sign (#) for each data buffer transferred over the network. The default is DISABLE TRANSFER_VERIFICATION.

UNIX Command

hash

Format

ENABLE TRANSFER_VERFICATION

Example

```
FTP> ENABLE TRANSFER_VERIFICATION
FTP> PUT X.TXT
---> PORT 130,180,4,10,4,165
200 PORT command successful.
---> STOR X.TXT
150 Opening data connection for X.TXT (130.180.4.10,1189).
#
226 Transfer complete.
2819 bytes sent in 00:00:00.11 seconds
```

Displays a number sign (#) for each data buffer transferred over the network.

EXIT

Closes any open connection with the remote server and returns you to the previous command level. Ctrl/Z has the same effect.

UNIX Command

```
quit
```

Format

EXIT

Example

```
FTP> EXIT
200 Goodbye.
```

Closes any open connection with the remote server.

GET

Copies one or more remote files from the remote host to the local host. If you do not specify a local file, the copied file is given the same name as the remote file.

You can also use the GET command to send files directly to a printer while in an FTP session. To use this feature, you must specify the actual name of the printer; you cannot use the queue name or logical name. Additionally, you cannot send multiple files to a printer, only individual files.

You can also use the GET command to transfer files between an FTP client and DECnet host, if the client and host are connected through a host running DEC TCP/IP software.

To copy multiple files from the remote VMS host to the local host, you can also use the `mget` command. The `mget` command is available through the UNIX-based command interface. For more information on selecting a command interface, see Chapter 4. For a more complete description of `mget`, see your UNIX documentation.

UNIX Command

`get` *remote_file* [*local_file*]

`mget` *remote_files*

Format

GET ["]*remote_file*["] [, | + ...] [*local_file*]

Command Qualifier	Default
/FDL	None.

Parameters

remote_file

A valid file specification indicating the file on the remote host to be copied to the local host. You can use the asterisk as a wildcard in the remote file specification.

You can specify multiple remote files by separating them with with commas (,) or plus signs (+). If you specify multiple remote files, you cannot specify a local file.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

local_file

A valid VMS file specification indicating the name of the copied file on the local host. If you do not specify a local file, the local file specification defaults to the supplied remote file specification. If the defaulted file specification string includes a device or directory they are removed.

If you use wildcards or specify multiple files in the remote file specification, you cannot specify a local file name.

Command Qualifier

/FDL

Creates a secondary file that contains the file's VMS record attributes. When a file and its accompanying FDL file are copied from UNIX to VMS, the FDL file is used to recreate the file with the correct attributes.

A transfer of ASCII-type data results in a sequential file with variable records; a transfer of IMAGE-type data results in a sequential file with fixed records of 512 bytes. ASCII is the default. The SET TYPE command determines the type.

Examples

1.
```
FTP> GET "myfile" MAIL.TXT
---> PORT 130,180,4,10,4,172
200 PORT command successful.
---> RETR myfile
150 Opening data connection for myfile (130.180.4.10,1194) (52 bytes).
226 Transfer complete.
53 bytes received in 00:00:00.01 seconds
```

   Copies the remote file `myfile` to the local host, where it is given the file name MAIL.TXT.

2.
```
FTP> GET NOTES.TXT, MAIL.TXT
---> PORT 130,180,4,10,4,178
200 PORT command okay.
---> RETR NOTES.TXT
150 Opening data connection for NOTES.TXT (130.180.4.8,1199)
226 Transfer complete.
959 bytes received in 00:00:00.02 seconds
---> PORT 130,180,4,10,4,176
200 PORT command okay.
---> RETR MAIL.TXT
150 Opening data connection for MAIL.TXT (130.180.4.10,1200)
226 Transfer complete.
14 bytes received in 00:00:00.01 seconds
```

   Copies the remote files NOTES.TXT and MAIL.TXT to the local host, where they are given the file names NOTES.TXT and MAIL.TXT.

3.
```
FTP> GET *.txt
```

   Copies all the files with .TXT as the extension from the remote default directory to the local default directory.

4.
```
FTP> mget *.txt
```

   Copies all the files with .txt as the extension from the remote default directory to the local default directory.

   Assumes that a connection to the remote host is open and that you are using FTP by way of the UNIX-based standard FTP command interface.

5.
```
FTP> GET/FDL "x.txt"
```

   Preserves the record attributes of the file `x.txt` during the copy.

6.
```
FTP> GET "q4earnings" LPA0
```

   Retrieves the file `q4earnings` and sends it directly to a printer (LPA0).

7. `FTP> GET MYNODE::DEVICE:[DIR]MEMO.TXT MEMO.TXT`

   Copies the file MEMO.TXT from a DECnet host to the local host.

HELP

Invokes the FTP help facility to display information about FTP commands.

UNIX Command

`help` [*command_name*]

`?` [*command_name*]

Format

HELP [*command_name*]

Command Qualifier	Default
/REMOTE	None.

Parameter

command_name

The name of the FTP command whose information is to be displayed.

Command Qualifier

/REMOTE

Specifies that the remote FTP server displays the requested help information. You must be connected to a remote FTP server to use this qualifier.

Examples

1. FTP> HELP LOGIN

```
LOGIN

    Used to initiate a login to a remote FTP server host.

    The user_name parameter is case sensitive on some types of
    systems.  Therefore, you must quote the user_name input
    if you want to send lower case characters to the remote host.

    FORMAT:
     LOGIN ["]user_name["]

Topic?
```

Displays information about the LOGIN command.

2. FTP> HELP/REMOTE

```
214-The following commands are recognized (* =>'s unimplemented).
USER    PORT    RETR    MSND*   ALLO    DELE    SITE*   XMKD    CDUP
PASS    PASV    STOR    MSOM*   REST*   CWD     STAT*   RMD     XCUP
ACCT*   TYPE    APPE    MSAM*   RNFR    XCWD    HELP    XRMD    STOU
REIN*   STRU    MLFL*   MRSQ*   RNTO    LIST    NOOP    PWD
QUIT    MODE    MAIL*   MRCP*   ABOR    NLST    MKD     XPWD
```

Displays the remote help information.

LOGIN

Lets you log in to an account on the remote server (host). You are prompted for the password on a separate line; the password is not echoed. If the user name and password are valid, FTP allows you to transfer the data.

UNIX Command

`user` *username*

Format

LOGIN ["]*username*["]

Parameter

username

The name of the user's account on the remote host. If the system you are logging in to is case sensitive, you must put the user name in quotation marks to preserve case sensitivity.

Examples

1.
```
FTP> LOGIN SMITH
331 Password required for SMITH.
Password:
230 User SMITH logged in.
```

   Logs in to the account SMITH on the remote host.

2.
```
FTP> LOGIN "smith"
331 Password required for smith.
Password:
230 User smith logged in.
```

   Logs in to the account `smith` on the remote host. Quotation marks are not used when you enter the password.

PUT

Copies one or more files from the local host to the remote host. If you do not specify a remote file, it defaults to the local file name.

If you use the VMS interface, the file names are transferred in uppercase with version numbers by default. You can change this by using the /NOVERSION and /LOWERCASE qualifiers, which have no effect when you specify a remote file name. Transfers can also be affected by the ENABLE PARSE and DISABLE PARSE commands.

If you are using the standard FTP interface commands, the file names are transferred in lowercase without version numbers by default.

You can also use the PUT command to send files directly to a printer while in an FTP session. To use this feature, you must specify the actual name of the printer; you cannot use the queue name or logical name. Additionally, you cannot send multiple files to a printer, only individual files.

You can also use the PUT command to transfer files between an FTP client and DECnet host, if the client and host are connected through a host running DEC TCP/IP software.

UNIX Command

`put` *local_file* [*remote_file*]

`mput` *local_files*

Format

PUT *local_file* [["] [, | + ...] *remote_file* ["]]

Command Qualifiers	Defaults
/FDL	None.
/LOWERCASE	See text.
/[NO]VERSION	/VERSION

Parameters

local_file

A valid VMS file specification on the local host that indicates the name of the file to be copied to the remote host.

You can specify multiple VMS files for the local file by separating the file names with commas (,) or plus signs (+).

You can use wildcards in the local file specification.

If you use wildcards or specify multiple files in the local file specification, you cannot specify a remote file name.

remote_file

A valid file specification indicating the file name to be given the file on the remote host. If you do not specify a remote file, the name defaults to the local file specification. If the defaulted file specification string has a device or directory specified they are removed You cannot use wildcards in the remote file specification.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

Command Qualifiers

/FDL

Creates a secondary file that contains the file's VMS record attributes. When a file and its accompanying FDL file are copied from UNIX to VMS, the FDL file is used to recreate the file with the correct attributes.

A transfer of ASCII-type data results in a sequential file with variable records; a transfer of IMAGE-type data results in a sequential file with fixed records of 512 bytes. ASCII is the default. The SET TYPE command determines the type.

/LOWERCASE

Specifies that all file names are converted to lowercase when transferred to the remote server. By default, file names are transferred in uppercase. The /LOWERCASE qualifier has no effect when you specify a remote file name.

/[NO]VERSION

Specifies whether the version number is removed when files are transferred to the remote server. The /[NO]VERSION qualifier has no effect when you specify a remote file name.

The /VERSION qualifier specifies that the file is to be transferred with a version number. If the file is transferred to a non-VMS system, the semicolon is converted to a period. This is the default.

The /NOVERSION qualifier specifies that the file is to be transferred without a version number.

Examples

1.
```
FTP> PUT LOCAL.TMP
200 PORT command successful.
150 Opening data connection for LOCAL.TMP.13 (130.180.4.10,1208).
226 Transfer complete.
local: local.tmp;13 remote: local.tmp;13
14 bytes sent in 00:00:00.04 seconds
```

   Copies the file LOCAL.TMP to a remote UNIX system. The file name is transferred in uppercase with the version number by default. The semicolon separator is converted to a period.

2.
```
FTP> PUT LOCAL.TMP "localtmp"
```

   Copies the file LOCAL.TMP to a remote UNIX system. The file is transferred without a version number and in lowercase, because that is how the remote file name is specified.

3.
```
FTP> PUT MEMO.TXT;*
```

   Copies versions of the file MEMO.TXT to the remote host in uppercase with version numbers. The semicolon separators are converted to a periods.

4.
```
FTP> PUT/LOWERCASE/NOVERSION MEMO*.TXT
```

   Copies the most recent version of all files that fit the criteria to the remote host in lowercase without version numbers.

5. `FTP> PUT Q4EARNINGS LPA0`

   Copies the file Q4EARNINGS directly to a printer (LPA0).

6. `FTP> PUT ARIEL::DEVICE:[DIR]MEMO.TXT MEMO.TXT`

   Copies the file MEMO.TXT from the local host to a DECnet host.

QUOTE

Sends arguments directly to the remote FTP server without parsing them locally. This allows you to use FTP commands that are implemented by the remote FTP server, but not by the local host.

You can obtain a list of the commands implemented on the remote FTP server by entering a HELP/REMOTE command.

You cannot use the QUOTE command to send data transfer commands.

UNIX Command

```
quote
```

Format

QUOTE *arg1 [arg2 ...]*

Parameter

arg

The name of the FTP command you want to execute on the remote FTP server.

Examples

1.
```
FTP> QUOTE PWD
257 "/usr/staff/r0123/name
```

   Shows the pathname of the current working directory on the remote host.

2.
```
FTP> QUOTE MKD TEMP
257 MKD command successful.
```

   Creates the directory TEMP on the remote host.

RENAME

Renames the specified file on the remote host.

UNIX Command

`rename` *oldname newname*

Format

RENAME ["] *oldname* ["] ["] *newname*["]

Parameters

oldname

A valid file name on the remote host to be given a new file name.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

newname

A new valid file name on the remote host.

Any file specification that does not conform to VMS file specification rules must be placed in quotation marks.

Example

```
FTP> RENAME "scoretext" COUNT.TXT
350 File exists, ready for destination name
250 RNTO command successful.
```

The remote file name `scoretext` is changed to COUNT.TXT.

SET DEFAULT

Sets the working directory on the remote or local host to the specified directory. If the remote host is a VMS system, you can set the default to a nonexistent directory.

Unless you use the /LOCAL qualifier, the directory on the remote host is affected by default.

UNIX Command

`cd` *directory_name*

`lcd` *directory_name*

Format

SET DEFAULT ["]*directory_name*["]

Command Qualifiers	Defaults
/LOCAL	None.

Parameter

directory_name
The name of the new working directory. The directory name must be compatible with directory specifications of the target system. If the directory specification is a non-VMS specification, the directory name must be enclosed in quotation marks.

Command Qualifier

/LOCAL
Specifies that the current working directory on the local host is to be changed to the directory specified.

Examples

1.
```
FTP> SET DEFAULT "/usr/temp"
250 CWD command successful.
FTP> SHOW DEFAULT
257 "/usr/temp" is current directory.
```

Changes the current working directory on the remote host to `/usr/temp` using the absolute pathname.

2.
```
FTP> SHOW DEFAULT
257 "/usr/temp" is current directory.
FTP> SET DEFAULT "docs"
250 CWD command successful.
```

Changes the current working directory on the remote host to `/usr/temp/docs` using the relative pathname.

3.
```
FTP> SET DEFAULT/LOCAL USER$1:[SMITH.COM]
Local Directory now USER$1:[SMITH.COM]
```

Changes the current working directory on the local host to USER$1:[SMITH.COM].

SET MODE

Defines the way data is to be transferred. STREAM is the default mode and is the only mode Applications supports.

UNIX Command

`mode` *mode_name*

Format

SET MODE *mode_name*

Parameter

mode_name

Specifies the file transfer mode. Applications supports only the STREAM mode.

Example

```
FTP> SET MODE STREAM
```

File transfer mode is set to STREAM.

SET STRU

Defines the structure of a file that is to be transferred. DEC TCP/IP Applications for VMS software supports FILE structure and RECORD structure. FILE structure is the default.

Note

Some systems do not support RECORD structure and a SET STRU command will have no effect. To specify RECORD structure when the server does not support it, do the following:

1. On the UNIX system, FTP to the VMS system.
2. Enter a `quote stru r` command. The `quote` command will send the `stru r` command to the FTP server on the VMS system. The server will set the structure to RECORD.

For example, To transfer a file on `abyss` (a UNIX system) in RECORD structure to FRED (a VMS system), enter the following commands:

```
abyss> ftp fred
ftp> quote stru r
ftp> put myfile
```

If you enter a DIR/FULL of myfile, the record format will be variable length.

UNIX Command

`struct` *stru_name*

Format

SET STRU *stru_name*

Parameter

stru_name
Either FILE or RECORD.

For ASCII type files:

- FILE specifies the default file structure as STREAM_LF where the files consist of a continuous sequence of data bytes.
- RECORD specifies the default file structure as variable length. RECORD structured files are sequential with variable records.

For an IMAGE type file, the file is a 512 byte fixed length record for either FILE or RECORD structure.

Example

```
FTP> SET STRU RECORD
FTP> GET MYFILE
FTP> DIR/FULL MYFILE
```

In this example, the local and remote hosts are VMS systems. The following sequence occurs:

- The file structure is set to RECORD.
- Copies MYFILE to local host.
- Shows record format field of MYFILE to be variable length.

SET TYPE

Defines the data representation type for the data transfer. The data representation types defined in FTP are ASCII and IMAGE.

ASCII type is the default and is appropriate for text files. IMAGE type is appropriate for transferring binary files, such as executable images.

UNIX Command

`type` *type_name*

Format

SET TYPE *type_name*

Parameter

type_name
The data representation type. ASCII and IMAGE are valid data representation types.

```
FTP> SET TYPE IMAGE
200 Type set to I.
```

Specifies the data representation type to be IMAGE.

```
FTP> SET TYPE ASCII
200 Type set to A.
```

Specifies the data representation type to be ASCII.

SHOW DEFAULT

Displays the name of the current working directory on the remote host.

UNIX Command

pwd

Format

SHOW DEFAULT

Command Qualifier	Default
/LOCAL	None.

Command Qualifier

/LOCAL

Displays the current working directory on the local host.

EXAMPLES

1.
```
FTP> SHOW DEFAULT
257 "/usr/staff/r1/smith" is current directory.
```

   Displays the name of the current working directory on the remote host.

2.
```
FTP> SHOW DEFAULT/LOCAL
Local directory is WORK3$:[SMITH].
```

   Displays the current working directory on the local host.

SHOW STATUS

Displays the current status for the FTP session.

UNIX Command

```
status
```

Format

SHOW STATUS

Example

```
FTP> SHOW STATUS
Connected to: remote
Mode = stream, Type = ascii, Form = non_print, Structure = file
```

Displays the current status of FTP.

SPAWN

Suspends the current FTP session and spawns a subprocess running the DCL command interpreter on the local VMS host.

UNIX Command

!

Note that the ! command is provided as the equivalent of the SPAWN command when you are using the standard FTP commands on a VMS system. In a UNIX environment, the equivalent command is !. .

Format

SPAWN

Example

```
FTP> SPAWN
$
```

Spawns a subprocess on the local VMS host.

A

TELNET Messages

This appendix provides information about TELNET error messages.

%UCX-E-TELNET_NORBUF, Buffer not available.

Explanation: The TELNET client does not have enough buffer space to read from the socket.

User Action: None.

%UCX-E-TELNET_CRSOCK, Failed to set up socket configuration.

Explanation: The TELNET client failed to create a socket to establish a connection with the remote host.

User Action: Check to see whether the network is up and running. If it is running, make sure that the socket limit has not been reached.

%UCX-E-TELNET_CONECT, Failed to connect to remote host.

Explanation: The TELNET client failed to establish a connection with the remote host.

User Action: Check to see whether the remote server is running.

%UCX-E-TELNET_DISCON, Disconnected network connection.

Explanation: The remote server disconnected the session.

User Action: Check to see whether the network and the remote server are running. Try to reestablish the connection.

%UCX-E-TELNET_GETHST, Error in getting host address.

Explanation: The TELNET client failed to get the address of the host it needed to connect to.

User Action: Check to see whether the host is defined in the host database.

%UCX-E-TELNET_NOMEM, Failed to allocate memory.

Explanation: The TELNET client failed to allocate memory for its internal data structures.

User Action: Increase the virtual memory available.

%UCX-E-TELNET_NOSESS, There are no active sessions.

Explanation: There were no active sessions when you entered the DISCONNECT command.

User Action: None.

%UCX-E-TELNET_REDERR, Failed to read from network.

Explanation: The TELNET client was unable to read from the socket.

User Action: Check to see whether the network is up and running.

%UCX-E-TELNET_SCKOPT, Failed to set socket options.

Explanation: The TELNET client failed to set socket options.

User Action: Check to see whether the network is up and running.

%UCX-E-TELNET_TRDERR, Failed to read from terminal.

Explanation: The TELNET client failed to read the data from the terminal.

User Action: None.

%UCX-E-TELNET_TWRTER, Failed to write to terminal.

Explanation: The TELNET client failed to write the data to the terminal.

User Action: None.

%UCX-E-TELNET_WRTERR, Failed to write on socket.

Explanation: The TELNET client failed to write the data to the socket.

User Action: Check to see whether the network is up and running.

B

File Transfer Protocol Messages

The following sections provide information about FTP messages.

B.1 FTP Server Error Messages

This section describes the FTP error messages that can be displayed on the server host.

%FTP-E-ACPTCN, Accepting connection failed.

Explanation: The maximum number of active device-sockets is under the control of the system manager. When the device-socket quota exceeds the limit set by the system manager, the establishment of a socket connection fails. This results in a failure to accept the data connection from the remote host.

User Action: Change the limit set for the device-socket quota. If you cannot change it, ask the system manager to change it.

%FTP-I-CHINFO, *process*

Explanation: This message is displayed when an error occurs in a child process.

User Action: Verify the associated error message.

%FTP-E-CONHST, Error connecting to remote host: *host*.

Explanation: This error may occur when there is a network error or when the network is not active.

User Action: Make sure that network communication is active.

%FTP-E-CRCHPR, Error in creating a child process *process*.

Explanation: The FTPD server has failed to create a child process. This error occurs when insufficient system dynamic memory is available or the maximum number of processes that can be created at one time has been exceeded.

User Action: Ask the system manager to increase the MAXPROCESSCNT system parameter or increase the size of the nonpaged pool.

%FTP-E-CREMBX, Error in creating a mailbox *mailbox*.

Explanation: The FTPD server failed to create the mailbox with which it communicates with the child process. This error occurs when insufficient system dynamic memory is available to complete the service.

User Action: Increase the size of the nonpaged pool.

%FTP-E-CREPRC, Error in creating a child process *value*.

Explanation: The FTPD server failed to create a child process. This error occurs when insufficient system dynamic memory is available to complete the service.

User Action: Increase the size of the nonpaged pool.

%FTP-E-CRSOCK, Error creating a socket. - *value*.

Explanation: The FTP sever failed to create an endpoint for communication. This error usually occurs if there is a network error, if network has not been started, or if the device-socket quota exceeds the limit set by the system manager.

User Action: Make sure network communication is active on the system. Change the limit set for the device-socket quota. If you cannot change it, ask the system manager to change it.

%FTP-E-ECNSES, Error in continuing session for *host*. Error:*value*.

Explanation: A problem occurred in the network that caused the session to close.

User Action: Make sure network communication is active on the system.

%FTP-E-ESTSES, Error in starting session for *host*. Error:*value*.

Explanation: The FTP server failed to start a session.

User Action: Make sure network communication is active on the system.

%FTP-E-EXQUOT, Exceeded session quota.

Explanation: The maximum number of sessions has been exceeded. You can set the maximum number of FTP sessions by editing the appropriate parameter in the SYS$MANAGER:UCX$FTPD_STARTUP.COM command procedure.

User Action: Wait for a session to become free.

%FTP-E-GETDVI, Error in getting device information.

Explanation: The FTP server failed to get the device information.

User Action: Check to see whether a mailbox was created.

%FTP-E-GETHST, Error in getting host name.

Explanation: FTP server failed to get the host information.

User Action: Check to see whether the host database exists.

%FTP-E-GPRNAM, Error in getting peer name.

Explanation: The FTP server failed to get the name of the remote system.

User Action: Make sure network communication is active on the system.

%FTP-E-GSCKNM, Error in getting socket name.

Explanation: The FTP server failed to get the socket name.

User Action: Make sure network communication is active on the system.

%FTP-E-LOGREJ, Login request rejected.

Explanation: The login to the remote host was rejected. An invalid user name or password results in this kind of failure. If you have no network access privilege, the login request is rejected.

User Action: Log in with a valid user name and password to see if you have network access privilege.

%FTP-E-MBXINF, Error in getting mailbox information *mailbox*.

Explanation: The FTPD server failed to get the mailbox information of the mailbox it created to communicate with the child process.

User Action: You should never see this error. If it occurs and you have SPR support, submit a Software Performance Report (SPR) that describes the conditions leading to the error.

%FTP-E-NETERR, I/O error on network device.

Explanation: This error usually occurs if there is a network failure. Most of the communication calls fail with this error.

User Action: Make sure network communication is active on the system.

%FTP-E-OPNINP, Error opening *filespec* for input.

Explanation: This error usually occurs if the specified file does not exist or if there is a file protection violation.

User Action: Verify the protection applied to the file and change the protection if necessary. If this error persists even with proper specification of the file and proper privileges, submit a Software Performance Report (SPR) that describes the conditions leading to the error, if you have SPR support.

%FTP-E-OPNOUT, Error opening for output: *filespec*.

Explanation: This error usually occurs if a privilege violation occurs while creating this file.

User Action: Verify the protection applied to the directory in which the file has to be opened for output. Ask the system manager for the necessary privileges, if needed.

%FTP-E-QIOERR, QIO failure.

Explanation: The QIO failed during a write or read operation or the setting of an AST.

User Action: Verify the status with the system manager.

%FTP-I-SESCON, FTPD: Session connection from *host* at *time*.

Explanation: When a connection is established, this message is logged stating the name of the client from which the connection came and when it came.

User Action: None.

%FTP-I-SESDCN, FTPD: Session disconnection from *host* at *time*.

Explanation: When a session is disconnected, this message is logged stating the name of the client from which it was disconnected and when it was disconnected.

User Action: None.

%FTP-E-SESEXT, *process* session exited due to an error at *time*.

Explanation: When a session is disconnected because of an error, this message is logged stating the name of the client from which it was disconnected and when it was disconnected.

User Action: None.

%FTP-E-SETCNF, Failed to set up socket configuration. Server exiting.

Explanation: The FTP server failed to create a socket and bind an address to it. This error occurs when the Internet driver is not active.

User Action: Start the Internet driver.

%FTP-W-SUPFRM, FTP supports only form: *form*.

Explanation: If the specified form is not supported, nonprint becomes the default form.

User Action: None.

%FTP-W-SUPMOD, FTP supports only mode: *mode*.

Explanation: If the specified mode is not supported, STREAM becomes the default mode.

User Action: None.

%FTP-W-SUPSTR, FTP supports only stru: *structure*.

Explanation: If the specified structure is not supported, FILE becomes the default structure.

User Action: None.

%FTP-E-UNKMOD, Unknown mode: *mode*.

Explanation: The specified data transfer mode is unknown.

User Action: FTP supports only STREAM mode. Determine whether the specified mode is supported.

%FTP-E-WRTERR, Write error on network device.

Explanation: The FTP server failed to write data to a network device. This might be due to network communication problems.

User Action: Determine whether the network communication is up and running properly. If the error persists, submit a Software Performance Report (SPR) that describes the conditions leading to the error.

B.2 FTP Client Error Messages

This section describes the FTP error messages that can be displayed on the client host.

%FTP-E-ACPTCN, Error accepting data connection from remote host.

Explanation: The maximum number of active device-sockets is under control of the system manager. When the device- socket quota exceeds the limit set by the system manager, establishing a socket connection fails. This in turn results in a failure to accept the data connection from the remote host.

User Action: Change the device-socket quota. If you cannot change it, ask the system manager to change it.

%FTP-E-CONHST/FAO=1, Error connecting to remote host: *host*.

Explanation: This error may occur when there is a network error or when the network is not active.

User Action: Make sure that network communication is active.

%FTP-E-CONCNC, Control connection is not set up to remote host.

Explanation: A remote command has been requested before a control connection has been established to that host.

User Action: First establish a connection with the remote host with the FTP command CONNECT.

%FTP-E-CRSOCK, Error creating a socket.

Explanation: The FTP client failed to create an endpoint for communication. This error is most commonly seen if there is a network error or if the network has not been started or when the maximum number of device-sockets exceeds the limit set by the system manager.

User Action: Make sure network communication is active on the system. Change the limit for the socket quota. If you cannot change it, ask the system manager to change it.

%FTP-E-DISCON/FAO=1, Disconnect first. Connection already exists to: *host*.

Explanation: The FTP client supports only one active control connection. It tried to establish another connection but failed.

User Action: Disconnect the connection with the FTP DISCONNECT command first.

%FTP-E-FDLGEN, Failed to generate an FDL file for *file*.

Explanation: The FTP client failed to generate an FDL file for the local file that has to be transferred. This error could be due to insufficient virtual memory or a fatal internal error in the run-time library.

User Action: Ask the system manager to increase the virtual memory available. Restart the transfer. If the error persists, submit a Software Performance Report (SPR) that describes the conditions leading to the error.

%FTP-E-GETHST, Error in getting information for host: *host*.

Explanation: The FTP client failed to get the host information.

User Action: Check the integrity of the host database.

%FTP-E-LOGREJ, Login request rejected.

Explanation: The FTP client failed to log in to the remote host. An invalid user name or password results in this kind of failure.

User Action: Log in with a valid user name and password to see if you have network access privileges.

%FTP-E-NETERR, I/O error on network device.

Explanation: This error usually occurs if there is a network failure. Most of the communication calls fail with this error.

User Action: Determine whether the network communication is up and running properly.

%FTP-NOFDLF, FDL File *file* does not exist on remote host.

Explanation: This occurs during the data transfer if the /FDL qualifier is specified and there is no FDL file on the remote host.

User Action: Determine whether the FDL file exists.

%FTP-E-OPNINP/FAO=1, Error opening *file* for input.

Explanation: This error should not occur unless the specified file does not exist or if there is a file protection violation.

User Action: Verify the protection applied to the file and change the protection if necessary. If this error persists even with proper specification of the file and proper privileges, submit a Software Performance Report (SPR) that describes the conditions leading to the error, if you have SPR support.

%FTP-E-OPNOUT/FAO=1, Error opening for output: *file*.

Explanation: This error should not occur unless a privilege violation occurs during the creation of this file.

User Action: Verify the protection applied to the directory in which the file has to be opened for output, and ask the system manager for the necessary privileges if needed.

%FTP-E-READER, Read error on network device.

Explanation: The FTP client failed to read the data that is being transferred over the network. This might be due to a network communication problem.

User Action: Determine whether the network communication is up and running properly. If the error persists, submit a Software Performance Report (SPR) that describes the conditions leading to the error.

%FTP-W-SUPMOD/FAO=1, FTP supports only *mode* mode.

Explanation: If the specified mode is not supported, STREAM becomes the default mode.

User Action: None.

%FTP-W-SUPFRM/FAO=1, FTP supports only *form* form.

Explanation: If the specified form is not supported, NONPRINT becomes the default form.

User Action: None.

%FTP-W-SUPSTR/FAO=1, FTP supports only *structure* stru.

Explanation: If the specified structure is not supported, FILE becomes the default structure.

User Action: None.

%FTP-E-UNKHST/FAO=1, Unknown host: *host*.

Explanation: You tried to establish a connection with an unknown host.

User Action: Check to see whether the specified host is in the host database.

%FTP-E-UNKMOD/FAO=1, Unknown mode: *mode*.

Explanation: The specified data transfer mode is unknown.

User Action: Check to see whether the specified mode is supported.

%FTP-E-UNKTYP/FAO=1, Unknown type: *type*.

Explanation: The specified data transfer type is unknown.

User Action: Check to see whether the specified type is supported. Check to see whether it is uppercase.

%FTP-E-WRTERR, Write error on network device.

Explanation: The FTP client failed to write data to the network device. This might be due to a network communication problem.

User Action: Determine whether the network communication is up and running properly. If the error persists, submit a Software Performance Report (SPR) that describes the conditions leading to the error, if you have SPR support.

%FTP-I-CONCTO/FAO=1, Connected to: *host*.

Explanation: When a connection is established, this message specifies the name of the host with which the connection was established.

User Action: None.

%FTP-E-BADDIR, Invalid Directory

Explanation: The specified directory is either not valid or protected.

User Action: Verify that the directory exists and if necessary, change the protection.

Glossary

absolute pathname

A pathname that starts with a slash (/) and specifies a file that can be found by starting at the root of the file system and traversing the file tree.

Abstract Syntax Notation One (ASN.1)

See **ASN.1.**

access control list (ACL)

An ACL consists of access control entries that grant or deny access to a particular system object.

ACK

A control bit (acknowledgment flag) in the TCP header that indicates that the acknowledgment number field is significant for this segment.

active port

A port which is bound to a process.

address mask

A 32-bit mask used to select bits from an IP address for subnet addressing. It selects the network portion of the IP address and one or more bits of the local portion.

address resolution

Conversion of an IP address into the corresponding Ethernet or FDDI address. Depending on the network configuration, resolution may require broadcasting on a local network. *See* **Address Resolution Protocol.**

Address Resolution Protocol (ARP)

A protocol that dynamically binds Internet addresses and Ethernet or FDDI addresses.

addressing

A function that ensures that different communicating systems in a computer network are correctly identified at all times. Addressing in the Transport layer provides a unique address to every transport service access point.

agent

(1) In the client-server model, the part of the system that performs information preparation and exchange on behalf of a client or server application. An agent is the initiator of a call.

(2) Management interface into an entity. The agent accepts directives (management requests) from a director and then performs operations.

(3) The portion of an entity that performs management functions. The agent provides the interface to network management and is responsible for monitoring and controlling the service element, according to management requests, received through its access interface.

(4) The part of an entity that provides the interface to network management. An entity's agent is responsible for monitoring and controlling the service element, according to management requests, received through its access interface.

agent access module

That portion of an agent responsible for the agent's end of a management access protocol.

agent access point

An instance of a connection between a director or a superior entity and an agent.

agent address

The address that specifies the information needed by a director to establish communications with the agent's management interface.

agent attributes

Attributes maintained by the agent. Attributes do not cross the internal management interface.

alias node identifier

An optional node name and address used by some or all nodes in a VAXcluster, allowing the VAXcluster to be treated as a single node.

alternate address notation

An Internet address notation that conveys the same information as the common notation, but consists of two parts: network and host.

API

See **Application Program Interface.**

application

(1) A process that performs a specific network function; for example, the SMTP application allows you to exchange mail with remote systems.

(2) A user-defined image for a special-purpose function.

Applications were called "objects" in Phase IV DECnet–VAX.

application entity

The caller of one or more protocol machines of the Application layer within an application process.

Application layer

(1) The topmost layer in the Internet Reference Model that provides such communication services as preparing the data and mapping the names to addresses.

(2) Layer 7 of the OSI model. This layer determines the interface of the system with the user.

(3) The layer of the OSI Reference Model that provides for distributed processing and access. This layer contains the applications and protocols that use the lower layers. OSI application programs are also part of this layer.

application process

(1) An element within a real open system that processes the information for a particular application.

(2) A process within the Application layer that has evolved from a user's application program and which uses any of the upper-layer services.

application-process title

See **AP title.**

application program interface (API)

Set of calling conventions defining how a service is invoked through a software package.

AP title

Application-process title. An identifier of an application process.

architecture

(1) A model that defines the relative positions of specific functions and the types of relationships that these functions can make.

(2) The grouping of functions within a layer that provides the basis for defining formal protocols within the layer. For example, the OSI Reference Model is a layered model, based on the concept of functional layers.

area

A group of systems that make up a single level 1 routing subdomain. The systems can run independently as a subnetwork.

area address

The concatenation of the IDP and LOC-AREA fields of an NSAP address. A system can have more than one area address, but the system making up an area must have at least one area address in common with each of its neighbors.

area router

A level 2 router; a node that can send and receive packets and route packets from one node to another within its own area and between areas.

area routing

Forwarding of packets from one area within a network to another area by using level 2 routers. Nodes in a network are grouped into areas for routing purposes. Routing in a multiple area is hierarchial, with one level of routing within an area (level 1 routing) and a second, higher level of routing between areas (level 2 routing).

ARP

See **Address Resolution Protocol.**

ASN.1

Abstract Syntax Notation One; ISO standard notation for abstract syntax formation as defined by ISO 8824. See also the basic encoding rules.

asynchronous transmission

The mode of transmission in which the time intervals between character transmissions differ. Each character is surrounded by start and stop bits to allow the receiving device to recognize the beginning and end of each character (also called **start-stop transmission**). Most common over terminal lines and lines connecting other inexpensive devices, such as personal computers. *Contrast with* **synchronous transmission**.

attribute

The controllable or observable part of an entity; variable that network managers and applications programmers can manipulate for optimal performance.

Berkeley Internet Name Domain (BIND)

A host name and address lookup service for the Internet network. The BIND service is implemented in a client-server model. The client software (implemented by DEC TCP/IP Services for VMS), is referred to as the resolver. The resolver allows client systems to obtain host names and addresses from servers rather than from locally hosted databases. As such, you can use the BIND service to replace or supplement the host address mapping provided by the local UCX$HOST file.

Berkeley Software Distribution (BSD)

A derivation of the original UNIX operating system developed by the Computer Systems Research Group of the Department of Electrical Engineering and Computer Science at the University of California at Berkeley. The ULTRIX operating system is based on the BSD version of UNIX.

best effort delivery

A characteristic of network technologies that do not provide reliability at link levels. Best effort delivery systems work well with the Internet because the Internet protocols assume that the Internet protocols IP and UDP provide best-effort delivery service to application programs.

binding

Defining an UNIX or VMS file system to be a part of the DEC TCP/IP file system.

Border Gateway Protocol (BGP)

An inter-autonomous system routing protocol used to exchange network reachability information between autonomous systems. BGP runs over a reliable transport protocol, currently TCP. BGP was designed to replace EGP.

bound port

A port is bound to a process by an I/O function specifying a port number and Internet address for the device-socket.

Bps

Bits per second. A measure of the rate of data transmission.

bridge

(1) A device that expands the extent of a LAN by connecting it to another LAN or physical link. For example, Digital Equipment Corporation's LAN Bridge 100 manages inter LAN traffic and selectively forwards packets to keep local traffic local. Only data destined for different LANs passes through the bridge and continues on to the appropriate remote destination. Thus the bridge improves performance by reducing traffic between LANs. A Data Link layer relay for interconnecting LANs, used to increase the maximum number of stations, maximum distance, and total available bandwidth.

(2) A device that connects two or more physical networks and forwards packets between them. Bridges can usually be made to filter packets (that is, to forward only certain traffic). Related devices are repeaters, which simply forward electrical signals from one cable to another, and full fledged routers, which make routing decisions based on several criteria. In OSI terminology, a bridge is a Data Link layer intermediate system.

(3) A device that links two homogeneous packet broadcast local networks. It accepts all packets from each network addressed to devices on the other, buffers them, and retransmits them to the other network.

(4) A router that connects two or more networks and forwards packets among them. Usually, bridges operate at the physical network level. For example, an Ethernet bridge connects two physical Ethernet cables and forwards from one cable to the other exactly those packets that are not local. Bridges differ from repeaters because bridges store and forward complete packets while repeaters forward electrical signals.

broadcast

(1) In networks, the capability of sending a single message simultaneously to many nodes, just as television programs are sent to multiple television sets.

(2) A packet delivery system where a copy of a given packet is given to all hosts attached to the network. Example: Ethernet.

(3) Simultaneous transmission of data to a number of stations. A packet delivery system that delivers a copy of a given packet to all hosts that attach to it is said to broadcast the packet. Broadcast may be implemented with hardware (for example, as in Ethernet) or with software (for example, as in Cypress).

broadcast address

An address that designates all entities within a domain (for example, network, Internet).

broadcast addressing

A type of multicast addressing in which all nodes receive a message simultaneously.

broadcast mask

A mask used to interpret the Internet address as a broadcast address.

buffer

A device or an area of memory used for temporary storage when transmitting data from one device to another to compensate for a difference in rate of data flow or in time of occurrence of events. Buffers are used on routing nodes to temporarily store data that is to be forwarded from one node to another.

buffering level

In network communications, the number of buffers provided at one time by the network software to handle data. Single buffering tends to be less efficient than multibuffering but uses less memory on the local system. Multibuffering provides better performance if sufficient memory is available. With multibuffering, a network can send or process several buffers of data in quick succession.

cache

A process of storing blocks in memory for future use; used to minimize physical transfer of data between mass storage devices and memory. Can also be a very fast memory used in combination with slower, large-capacity memories.

CFSRTL

A VMS run-time library (RTL) that is used by the NFS server process to process files in the DEC TCP/IP file systems.

checksum

(1) An error-detecting code based on a summation operation performed on the A bits to be checked.

(2) A small, integer value computed from a sequence of octets by treating them as integers and computing the sum. A checksum is used to detect errors that result when the sequence of octets is transmitted from one machine to another. Typically, protocol software computes a checksum and appends it to a packet when transmitting. Upon reception, the protocol software verifies the contents of the packet by recomputing the checksum and comparing it to the value sent. Many Internet protocols use a 16-bit checksum computed with ones complement arithmetic with all integer fields in the packet stored in network byte order.

(3) A parameter that is carried in a block of data, and whose value can be used to determine whether or not the data was corrupted during transmission.

client

(1) A process that sends a request and waits for the results from another process (a server) that offers a service over the network.

(2) Another term for an access system.

client-server relationship

A model in which a client process sends a request and waits for the results from a server process. *See also* **client** and **server.**

common address notation

The common way of expressing an Internet address. The 32-bit address uses four fields that are separated by periods; each field ranges from 0 to 255.

connection

A logical communication path between two different processes.

container file

An RMS data file that contains a UNIX directory structure and UNIX file attributes for a UNIX file system. Each UNIX regular file is stored as a separate RMS data file using a system-assigned valid file name. The directory data files in the container file contain the UNIX file names and a pointer to the corresponding Files–11 data file.

Container File System (CFS)

The software that implements the container file structure. *See* **container file**.

control cluster

A group of small (256 bytes) buffers dynamically allocated from nonpaged pool memory. Control clusters store information related to device-sockets, internal control structures, Internet addresses, Internet routes, and Internet packet headers.

data cluster

A group of large (1792 bytes) buffers that store data in the system space. Transmit and receive operations service user processes by moving data to and from the data clusters.

datagram

A self-contained package of data carrying enough information to be routed from source to destination without reliance on earlier exchanges between source and destination or the transporting network.

datagram fragment

Datagram fragments are the result of fragmenting a datagram. The fragment carries a portion of data from the larger original and a copy of the original datagram header. The header fragmentation fields are adjusted to indicate the fragment's relative position within the original datagram.

datagram service

A datagram that is delivered in such a way that the receiver can determine the boundaries of the datagram as it was entered by the source.

DCL (VMS)

Digital Command Language. The standard command interface of the VMS operating system.

DEC NFS VMS server

A computer system that offers services to NFS clients within an Internet network environment. The computer system can be a single host, a whole VAXcluster system, or members of a VAXcluster system.

destination address

An Internet address that specifies where a datagram has to be sent. It contains the network, host identifiers, and the subnet identifier.

destination port

A 2-octet value in the TCP and UDP header field that identifies the destination upper-level protocol for a segment's data.

device driver

Software associated with each physical device in the system that serves as the interface between the operating system and the device controller.

device-socket

An extension of the pseudodevice, used for communications. The device-socket consists of the Internet pseudodevice and the socket.

Digital Management Control Center (DECmcc)

Management system software for monitoring, controlling, and testing entities in DECnet, DECnet/OSI, and multivendor distributed environments.

domain

(1) In the Internet, a part of a naming hierarchy. Syntactically, an Internet domain name consists of a sequence of names (labels) separated by periods (dots), for example, `tundra.mpk.ca.us`.

(2) In OSI, "domain" is generally used as an administrative partition of a complex distributed system, as in MHS Private Management Domain (PRMD) and Directory Management Domain (DMD).

dotted decimal notation

The syntactic representation for a 32-bit integer that consists of four 8-bit numbers written in base 10 with periods (dots) separating them. Used to represent IP addresses in the Internet as in: 192.67.67.20. Many Internet application programs accept dotted decimal notation in place of destination machine names.

Ethernet

Commonly used to refer to a local area network, an Ethernet itself is a passive coaxial cable; connections made to this cable contain the active components for communications.

event

A network- or system-specific occurrence for which the logging component maintains a record.

A measurable occurrence on a network that can be logged for later reference. For example, the arrival of a Protocol Data Unit (*See* **PDU**.)

exported file

A file with a UNIX file system that has been copied or linked into a VMS file system.

exporting a file system

Identifying a DEC TCP/IP file system or directory that can be remotely mounted by NFS clients.

FDDI

See **Fiber Distributed Data Interface.**

fetch-store operation

A paradigm that originates from the HELM management protocol described in RFCs 1021-1024. The concept makes it possible to do work with just two commands that allow a system manager to fetch a value from a data item or to store a value into a data item.

Fiber Distributed Data Interface (FDDI)

(1) A standard based on fiber optics, established by the American National Standards Institute (ANSI). FDDI provides a 100 mbps data rate using 1300 nanometer light wavelength. FDDI networks are limited to approximately 200 km in length, with repeaters every 2 km or less.

(2) An emerging high-speed networking standard; the underlying medium is fiber optics, and the topology is a dual-attached, counter-rotating token ring.

file specification

System-specific information that identifies a file on its storage system.

file system

A method for recording, cataloging, and accessing files on a volume.

Files–11 ODS level 2 structure

The set of rules that govern the organization of the VMS file system, external to the files themselves.

File Transfer Protocol (FTP)

A protocol that allows users to log in to a remote host, identify themselves, list remote directories, copy files to or from the remote host, and execute some commands remotely.

flow control

(1) A function performed by a receiving entity to limit the amount or rate of data that is sent by a transmitting entity.

(2) Control of the rate at which hosts or gateways inject packets into a network or Internet, usually to avoid congestion. Flow control mechanisms can be implemented at various levels. Simplistic schemes, like ICMP source quench, simply ask the sender to cease transmission until congestion ends. More complex schemes vary the transmission rate continuously.

(3) Flow control allows communicating layers to match their data transfer and receive rates. This ensures that one end of the connection does not send data faster than the other end can process it.

(4) A mechanism that keeps traffic within limits acceptable by the end receiver or any intermediate receiver. At the terminal level, the flow control mechanism must guarantee that the flow of characters will stop if the buffer fills up.

fragment

One of the pieces that results when an Internet gateway divides an IP datagram into smaller pieces for transmission across a network that cannot handle the original datagram size. Fragments use the same format as datagrams; fields in the IP header declare whether a datagram is a fragment and if so, the offset of the fragment in the original datagram. IP software at the receiving end must reassemble fragments into complete datagrams.

fragmentation

The breaking up of Internet datagrams into smaller datagrams. This allows a datagram originating in a network that allows a large packet size to traverse a network that limits packets to a smaller size. The Internet fragments are reassembled at the destination host.

FTP

See **File Transfer Protocol.**

function code

A parameter in a $QIO system service call that defines the specific function to be performed by that $QIO.

gateway

A host computer that interconnects two networks and transfers packets from one network to another.

hardware address

The low-level address used to specify the connection between the network controller of a node and the network cable. Ethernet uses 48-bit addresses assigned by the vendor. Address formats vary among network vendors. *See also* **Internet address.**

hard link

A hard link to a file is indistinguishable from the original directory entry. *See* **link**.

header

A collection of control information transmitted along with data between peer entities.

heterogeneous computer environment

A group of computer systems running different operating system software, such as VMS and UNIX.

hop

The logical distance between two adjacent nodes in a network. The logical distance between two nodes. One hop is the distance from one node to an adjacent node.

hop count

A time-to-live counter contained in each IP datagram that prevents datagrams from circling forever in the Internet due to exceedingly long or multiple-routing cycles.

host

A computer that acts as a source or destination of messages within the communication subnetwork (referred to as a node in VMS terminology).

host database

An Internet database that allows users to use host names. The database contains host names, Internet addresses of the hosts, and any alias names for the hosts.

Host-to-Host Communication layer

The second-highest level in the Internet architecture model. This layer provides end-to-end communication services, including mechanisms such as end-to-end reliability and network control. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) reside in this layer.

ICMP

See **Internet Control Message Protocol.**

IHL (Internet header length)

An IP header field that indicates the number of 32-bit words making up the Internet header.

imported file

A file within a VMS file system that has been copied or linked into a UNIX file system.

inetd

A server process that listens for requests for specific services. When inetd receives a request for a service, it starts the specified service.

INS (initial sequence number)

The first sequence number used for sending or receiving on a connection.

interface

The boundary between two parts of a system across which communication is possible. The interface may be defined through hardware or software.

Internet

(1) The largest Internet in the world, consisting of large national backbone networks (such as MILNET, NSFNET, and CREN) and a myriad of regional and local campus networks all over the world. The Internet uses the Internet protocol suite. To be on the Internet, you must have IP connectivity, that is, be able to TELNET to or ping other systems. Networks with electronic mail connectivity only are not actually classified as being on the Internet.

(2) The collection of networks and gateways, including the ARPANET, MILNET, and NSFnet, that use the TCP/IP protocol suite and function as a single, cooperative virtual network. The Internet provides universal connectivity and three levels of network services: unreliable, connectionless packet delivery; reliable, full duplex stream delivery; and application level services like electronic mail that build on the first two. The Internet reaches many universities, government research labs, and military installations.

Internet address

The low-level address used to specify the connection between the network controller of a node using TCP/IP and the network cable. The 32-bit address is composed of two parts: network number and host number.

Internet architecture

A four-layered communications model that consists of the following: Application layer, Host-to-Host Communication layer, Internet Protocol layer, and Network Protocol layer.

Internet Control Message Protocol (ICMP)

A special-purpose protocol that gateways use to communicate with the network software in hosts.

IP

See *Internet Protocol*

IP address

See **Internet address.**

IP datagram

(1) The unit of information passed across the Internet. Contains source and destination addresses, the data, and fields that define the length of the datagram, the header checksum, and flags to say whether the datagram can be (or has been) fragmented.

(2) The basic unit of information passed across the Internet. An IP datagram is to the Internet what a hardware packet is to a physical network. It contains a source and destination address along with data.

Internet Protocol (IP)

(1) A protocol that resides in the Internet layer and performs two major functions: Internetwork addressing and fragmentation of messages.

(2) Defines the Internet datagram as the unit of information passed across the Internet and includes ICMP control and error message protocol. The Internet protocol suite is often referred to as TCP/IP because IP is one of the two fundamental protocols.

IP trailer protocol

A protocol in which the protocol header follows the data.

Kbps

Kilobits Per Second. A measure of data transmission rate.

layer

(1) A grouping of related communication functions that provide a well defined service to a client independently of the protocols and other means used to provide it.

(2) In networks, layers pertain to the software protocol levels that make up the architecture. Each layer performs certain functions for the layers above it.

(3) An independent and self-contained set of interconnected functions with its own characteristic purpose and protocols within the OSI Reference Model.

(4) A clearly defined part of a network architecture that performs a set of related functions.

link

A directory entry referring to a file. A file may have several links to it. A hard link to a file is indistinguishable from the original directory entry. A file cannot be deleted (removed) until the link count is 0.

local address

The address of a host within a subnetwork.

local area network (LAN)

(1) A high-speed data communications network that covers a limited geographical area, such as an industrial complex or college campus.

(2) A network that provides high-speed communication in a moderate-sized geographical area. A general-purpose local network that can serve a variety of devices. Typically used for terminals, microcomputers, and minicomputers.

(3) Any physical network technology that operates at high speed (usually tens of megabits per second through several gigabits per second) over short distances (up to a few thousand meters). Examples include Ethernet and proNET-10.

(4) A local area network. A privately owned data communications system that connects information processing equipment. The geographical area is usually limited to a group of buildings, a single building, or a section of a building.

local network

The network directly attached to a host or gateway.

local subnetwork

The subnetwork directly attached to a host or gateway.

loopback

A test of connectivity to a specific host in the network.

Mail Exchanger

See MX.

Master File Directory (MFD)

The root of a DEC NFS VMS file system.

memory buffers (MBUFs)

Portions of memory that act as queues for data arriving at a port before the process is ready to claim that data.

MIB II

Management Information Base II. A collection of objects that can be accessed by a network management protocol. In this book, the database maintained by a gateway running SNMP.

multihomed host

A host computer that has two or more hardware connections to a network. Multihomed hosts require multiple Internet addresses.

MX

DEC TCP/IP implementation of the Mail Exchanger. MX allows nodes in a local network to forward mail to nodes that may not be directly connected to the local network.

network class

Defines the type of network addressing scheme being used. The high-order bits in the network number designate the network class of the Internet address.

network database

A database that allows users to refer to networks by name rather than network number. The database contains network names, Internet addresses for the networks, and any alias names for the networks.

Network File System (NFS)

(1) A distributed file system that allows a set of computers to cooperate in accessing each other's files transparently.

(2) In DEC TCP/IP Services, a server that provides transparent remote access to files on a VMS server for disk-based UNIX clients.

Network layer

The lowest layer in the Internet architecture model. This layer provides the mechanism for connecting the hosts to the networks.

network management

See **MIB II** and **Simple Network Management Protocol (SNMP)**.

network mask

A mask used to determine the subnetwork in the Internet address. Each bit that is turned on (binary one) in the mask is interpreted as part of the network and subnetwork address. Synonymous with subnet mask.

NFS

See **Network File System.**

NFS server

A VMS image that simultaneously processes multiple service requests from many NFS clients. The services provided are MOUNT, NFS, and PORTMAPPER.

nobody

A UNIX convention, used when file ownership is not known, which maps to an account with a UID and GID of –2.

NOC

(1) Network Operations Center. Any center tasked with the operational aspects of a production network. These tasks include monitoring and control, trouble-shooting, and user assistance.

(2) In our SNMP implementation model, this is where the network management stations reside that use SNMP agents to execute network management applications.

node

A system on a network. Also referred to as a host.

node address

Required unique numeric identification of a specific node in the network.

node name

The alphanumeric identification associated with the node address for a system in a one-to-one mapping.

pathname

A UNIX pathname is composed of a series of fields separated by slashes (/). Each field designates a file name that is uniquely contained in the previous field (directory).

ping

Packet Internet Groper. A program used to test reachability of destinations by sending them an ICMP echo request and waiting for a reply. The term is used as a verb: "Ping host X to see if it is up!"

port

The endpoint of a communication link between two processes.

privileged port

A port in which the remote host has done some level of checking against the application using the port. Privileged port numbers range from 1 to 1023.

protocol

A set of rules that controls the communications between computers. Also, a set of conventions between communicating processes regarding the format and contents of messages to be exchanged.

Protocols can describe low level details of machine-to-machine interfaces (for example, the order in which the bits from a byte are set across a wire), or high level exchanges between applications programs (for example, the way in which two programs transfer a file across the Internet).

proxy

Mechanism whereby one system representsr another system in responding to protocol requests. DEC TCP/IP uses a proxy mechanism to provide a VMS identity (account) for each UNIX client by adding the name and identification codes of the client to a proxy database.

pseudodevice

A pseudodevice is a software device used to implement special purpose transports and is not directly associated with hardware. A DEC TCP/IP pseudodevice is a data structure used by the BG driver in the I/O channel to interface with the Internet protocols.

reassembly

The process of piecing together datagram fragments to reproduce the original large datagram. Reassembly is based upon the fragmentation data in the IP header of the datagram.

Record Management Services (RMS)

The data management subsystem on VMS that defines the rules that govern the internal organization and the methods of accessing file data. VMS RMS, together with ODS-2, defines a set of rules that govern files in a VMS File System. These rules include how files are named and how files are cataloged in directories.

reliability

The ability of a protocol to recover data that is damaged, lost, duplicated, or delivered out of order.

relative pathname

A pathname that does not start at the root. The default directory is merged with the relative pathname to form the absolute pathname.

reserved port

An assigned port that provides services to unknown callers by providing a service contact point. Reserved port numbers range from 1 to 255.

RFC

Request For Comments. The document series, begun in 1969, which describes the Internet suite of protocols and related experiments. Not all (in fact very few) RFCs describe Internet standards, but all Internet standards are written up as RFCs.

RIP

See **Routing Information Protocol.**

rlogin

Remote login. The service offered by Berkeley 4.3 BSD UNIX systems that allows users of one machine to connect to other UNIX systems across the Internet and interact as if their terminals connected the machines directly. Although rlogin offers essentially the same service as TELNET, it is superior because the software passes information about the user's environment (for example, terminal type) to the remote machine.

RMS

See **Record Management Services.**

root

The top level directory in a UNIX file system.

In UNIX, the term "root" is also used to indicate a user (the superuser) who has special privileges. *See* **superuser.**

root mode

The file protection placed on a container file when it is created.

root name

The element of a path name that identifies the target file system.

route

The path over the network that information takes to get from one host to another.

router

A node that can send and receive data and also forward data to other nodes.

routing database

A database that contains routing information. The database includes destination host names, Internet addresses for the hosts, gateway host names, and Internet addresses for the gateways. There are two logical route databases: the static route database that is maintained on disk and the volatile database in memory.

Routing Information Protocol (RIP)

A protocol that enables gateways to broadcast their current routing database to hosts and networks that are connected directly to them. DEC TCP/IP Services for VMS software implements the RIP protocol through its dynamic routing server.

segment

The unit of data exchanged by the TCP modules.

segment length

The amount of sequence number space occupied by a segment, including any controls which occupy sequence space.

sequence number

A 32-bit field in the TCP header that contains the sequence number of a sequenced control flag, the first byte of data, or empty segments (the sequence number of the next data octet to be sent).

server

A process that offers a service to another process over the network. A server accepts requests from other processes, known as clients.

SMTP

(1) Simple Mail Transfer Protocol. The Internet electronic mail protocol. Defined in RFC 821, with associated message format descriptions in RFC 822.

(2) The Internet standard protocol for transferring electronic mail messages from one machine to another. SMTP specifies how two mail systems interact and the format of control messages they exchange to transfer mail.

Simple Network Management Protocol (SNMP)

The network management protocol of choice for TCP/IP-based Internets. The SNMP agent allows you to remotely monitor the Internet protocols by requesting information about Management Information Base (MIB-II) variables, which are described in RFC 1158.

socket

(1) The endpoint of communication that an Internet address and port may be bound to.

(2) A data structure that is part of the Internet pseudodevice that is created every time a VMS process assigns a communication channel. The other part of the Internet pseudodevice is the device-socket.

soft link

A pointer that provides an alternate name for an object or directory in the namespace. You can restructure a namespace on a minor scale by creating soft links that point from an existing name to a new name. For example, suppose an organization named parts of its directory structure based on functional groups, and the namespace administrators had a directory named `.admin.marketing` that they wanted to change to `.admin.sales`. They could create a soft link named `.admin.marketing` that pointed to the name `.admin.sales`. Soft links can also be a way to give a file more than one name, so that different kinds of users can refer to a name in a way that makes the most sense to them. A soft link can be permanent or it can expire after a period of time you specify.

source

An IP header field that contains the Internet of the datagram's point of origin.

source port

A 2-octet value in the TCP or UDP header field that identifies the source's upper-level protocol of a segment's data.

subnetwork

Subnetworking allows for organizing hosts within a network into logical groups. A network can be made up of several subnetworks.

subnet address

Part of the Internet addressing scheme. A site uses a single Internet address for multiple physical networks and interprets the local part of an address by dividing the address into the physical network part and the host part.

subnet mask

A mask used to determine the subnetwork in the Internet address. Each bit that is turned on (binary one) in the mask is interpreted as part of the network and subnetwork address. Synonymous with network mask. *See* **address mask.**

superuser

A superuser is a user who has been granted special privileges. A superuser has an effective UID of 0.

symbolic link

A symbolic link is a special type of file that contains the name of the file to which it is linked. The referenced file is used whenever the symbolic link file is opened.

Transmission Control Protocol (TCP)

The Internet standard transport level protocol that provides the reliable, full-duplex, stream service on which many application protocols depend. TCP allows a process on one machine to send a stream of data to a process on another. It is connection-oriented in the sense that before transmitting data, participants must establish a connection.

TCP/IP

The suite of protocols that make up the Transmission Control Protocol and the Internet Protocol. Transmission Control Protocol (TCP) is the Internet standard transport level protocol. Internet Protocol (IP) is the Internet standard protocol that defines the Internet datagram as the unit of information passed across the Internet. TCP and IP are the two most fundamental Internet protocols. *See* **Transmission Control Protocol** and *Internet Protocol*

TELNET

The Internet standard protocol for remote terminal connection service. TELNET allows a user at one site to interact with remote timesharing systems at another site as if the user's terminal connected directly to the remote machine. That is, the user invokes a TELNET application program that connects to a remote machine, prompts for a login ID and password, and then passes keystrokes from the user's terminal to the remote machine and displays output from the remote machine on the user's terminal.

TELNET protocol

The TELNET protocol enables users to access any system on your network running the TELNET server software and establishes a virtual terminal connection between the users' terminals and the specified hosts.

thread

A request from an NFS client to the NFS server.

`timeo`

The time out option for the NFS `mount` command.

User Accounting File (UAF)

The file that contains account names and their associated attributes.

UDP

See **User Datagram Protocol.**

UFS

A DEC NFS UNIX-style file system that is created using UCX commands and which resides on the VMS system.

ULTRIX file system

A collection of files organized as a tree with a single root node called the root that is written as a slash (/). Non-leaf nodes of the tree are directory files and leaf nodes of the tree are either directory or regular data files. On a Files–11 ODS-2 formatted disk, an ULTRIX file system is represented as a set of Files–11 files.

UNIX to UNIX Copy Program

See **UUCP.**

User Datagram Protocol (UDP)

An unreliable delivered protocol that depends on the underlying Internet Protocol to transport UDP messages from one host to another. Each UDP message contains the data sent by a user process, a destination port number, and a source port number.

UUCP

UNIX to UNIX Copy Program; allows one UNIX system to copy files to or from another UNIX system.

VAXcluster alias

A special alias that allows remote hosts to address the cluster of hosts as a single host as well as any cluster member individually.

VFS

A DEC NFS VMS file system. That is, a VMS-style file system created using UCX commands that resides on the VMS system.

VMS file system

The VMS files and directories on a mounted VMS volume. These files and directories reside on a Files–11 On-Disk Structured (ODS-2) disk.

window

A 2-octet field in the TCP header indicating the number of data octets (relative to the acknowledgment number in the header) that the sender is currently willing to accept.

Index

F

G

H

I

S

T

U

V

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-baud modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	————	Local Digital subsidiary or approved distributor
Internal[1]	————	USASSB Order Processing - WMO/E15 *or* U.S. Area Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

[1]For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

DEC TCP/IP Services for VMS Use

AA–PC27B–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less _____

_____

_____

What I like best about this manual is _____

_____

_____

What I like least about this manual is _____

_____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

_____

_____

_____

_____

I am using **Version** _____ of the software this manual describes.

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

---- Do Not Tear - Fold Here and Tape ----

digital™

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Telecommunications and Networks
LKG1–3/L12
550 KING STREET
LITTLETON, MA 01460–1289

---- Do Not Tear - Fold Here ----